1990-12

Sliding control design and implementation on a single-link flexible arm

Feng, Chieh-Chuan

Monterey, California: Naval Postgraduate School

(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SLIDING CONTROL DESIGN AND
IMPLEMENTATION
ON
A SINGLE-LINK FLEXIBLE ARM

by

Chieh-Chuan Feng

December, 1990

Thesis Advisor: Liang-Wey Chang

Approved for public release; distribution is unlimited.

Unclassified

Security Classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification: Unclassified	1b Restrictive Markings
2a Security Classification Authority	3 Distribution Availability of Report: Approved for public release; distribution is unlimited.
2b Declassification/Downgrading Schedule	
4 Performing Organization Report Number(s)	5 Monitoring Organization Report Number(s)
6a Name of Performing Organization: Naval Postgraduate School; 6b Office Symbol (If Applicable): 69	7a Name of Monitoring Organization: Naval Postgraduate School
6c Address (city, state, and ZIP code): Monterey, CA 93943-5000	7b Address (city, state, and ZIP code): Monterey, CA 93943-5000
8a Name of Funding/Sponsoring Organization; 8b Office Symbol (If Applicable)	9 Procurement Instrument Identification Number
8c Address (city, state, and ZIP code)	10 Source of Funding Numbers: Program Element Number; Project No; Task No; Work Unit Accession No

11 Title (Include Security Classification): SLIDING CONTROL DESIGN AND IMPLEMENTATION ON A SINGLE-LINK FLEXIBLE ARM

12 Personal Author(s): Chieh-Chuan Feng

13a Type of Report	13b Time Covered	14 Date of Report (year, month,day)	15 Page Count
Master's Thesis	From To	December 1990	130

16 Supplementary Notation: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms (continue on reverse if necessary and identify by block number)
Field	Group	Subgroup	SLIDING MODE CONTROL, FLEXIBLE ARM, CONTROL OF FLEXIBLE ARM, UNCERTAINTY

19 Abstract (continue on reverse if necessary and identify by block number

The sliding mode control is known as a robust control that is able to work under the uncertainties of modeling error and the environmental disturbances. The objective of this research is to design the simple control (sliding control) algorithms for a single-link flexible arm and to study the robustness due to varying payload. A general form of physical plant in state space is formulated. To achieve a continuous control, a time-varying boundary layer was introduced into the control system neighboring the sliding surface. The computer simulation program was coded in MATLAB. A low-cost IBM-AT micro-computer was utilized to implement the sliding control on the flexible arm.

20 Distribution/Availability of Abstract	21 Abstract Security Classification
[X] unclassified/unlimited [ ] same as report [ ] DTIC users	Unclassified

22a Name of Responsible Individual	22b Telephone (Include Area code)	22c Office Symbol
Liang-wey Chang	(408) 646-2632	MECk

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted

All other editions are obsolete

security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

Sliding Control Design and Implementation on a *Single-Link Flexible Arm*

by

Chieh-Chuan Feng

Lieutenant, Republic of China Navy

B.S., Chung-Cheng Institute of Technology , 1982

Submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

AND

MECHANICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL

December 1990

Author: [redacted]

Chieh-Chuan Feng

Approved By: [redacted]

Liang-Wey Chang, Thesis Advisor

[redacted]

Anthony J. Healey, Chairman

Department of Mechanical Engineering

[redacted]

DEAN OF FACULTY
AND GRADUATE STUDIES

ABSTRACT

The sliding mode control is known as a robust control that is able to work under the uncertainties of modeling error and the environmental disturbances. The objective of this research is to design the simple control (sliding control) algorithms for a single-link flexible arm and to study the robustness due to varying payload. A general form of physical plant in state space is formulated. To achieve a continuous control, a time-varying boundary layer was introduced into the control system neighboring the sliding surface. The computer simulation program was coded in MATLAB. A low-cost IBM-AT micro-computer was utilized to implement the sliding control on the flexible arm system.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENT

First, I would like to give my thanks to Jesus Christ, my Lord, with his strengthening, I can have my thesis completed, and then, I wish to express my deepest gratitude to my advisor, Professor Liang-Wey Chang, for his unfailing guidance and full support for this research. Last but not least, I wish to thank my wife, Roselinda, and my son, Fourier, for their patience and help during my studies.

I. INTRODUCTION

A. MOTIVATION

The control of flexible arms has been an active research area and a challenge to researchers. The motion control of flexible arms concerns the tip position and the control algorithm should be able to deal with the large motion and the small motion due to vibrations. It is difficult to have a precise arm model and an error exists between the model and the plant, that is modeling error. The modeling error for flexible arm may contain changing payload, high-frequency unmodeled dynamics, deviation on load position in the end-effector, and environmental disturbances.

As the operation of control system is concerned, the computation of the control law plays an important role. While the time-delaying may worsen the control system, the control structure has to be as simple as possible to reduce the computation time. Since the models of flexible arms are complicated, simplified models will be utilized and controllers will be designed accordingly.

Therefore, the challenge that we encounter in this research is to design simple control algorithms such that the modeling errors due to changing payload are compensated and the on-line operation will also be achieved on the single-link flexible arm. An IBM-AT is chosen for the low-cost implementation in this research.

B. LITERATURE REVIEW

The single-link flexible arm model of using Equivalent Rigid Link System (ERLS) was first derived by Chang [Ref. 1]. The ERLS described the motion in large motion and small

motion. A Sequential Integration Method [Ref. 2] was also derived to facilitate an effective integration routine. Petroka [Ref. 3] experimentally validated the ERLS dynamic model of single-link flexible arm which was built and driven by an electrohydraulic actuator. Gannon [Ref. 4] upgraded the model by using the natural-mode shape function. Park [Ref. 3] designed and simulated a closed-loop non-robust controller for the arm. Kirkland [Ref. 5] redefined and implemented the controller on an IBM-AT computer. A strain gage and a potentiometer were used to determine the tip position of the arm.

As the robust control algorithm is concerned, the Sliding Mode Control (SMC) [Ref. 6-9] has been known as a robust control that is insensitive to modeling error and disturbances. Figure 1 illustrates a synthesis view of the sliding mode control. The external disturbances and parameters variation will first be filtered by the S dynamics to produce S and the generated S will be fed into error dynamics to perform further filtering. Thus, the S dynamics play very important roles to provide robustness to the control system. Once the S dynamics reject all the unwanted signals, the error dynamics will present the system behavior. Because of the undesirable high speed switched control on the sliding mode, a boundary layer thickness was introduced to the control system and a smoothed control was designed. Fan [Ref. 10] simulated and implemented the sliding mode control on the single link flexible arm. The robustness of the control was proved despite of a simplified model. Straight sliding control [Ref. 11] and versatile sliding control [Ref. 12] utilizing the idea of filtering provide more tuning capability on the S dynamics such that the unwanted signals will be filtered. In this research, these sliding control algorithms will be applied to the single link flexible arm to study the control system dynamics.

C. OBJECTIVE

The objective of this research is to utilize simple control structures (sliding mode controls) to perform the motion control of a single-link flexible arm. The robustness due to varying payload will be studied. The control design will be performed in a state-space form using matrix-norm techniques. An IBM-AT will be chosen for the implementation.

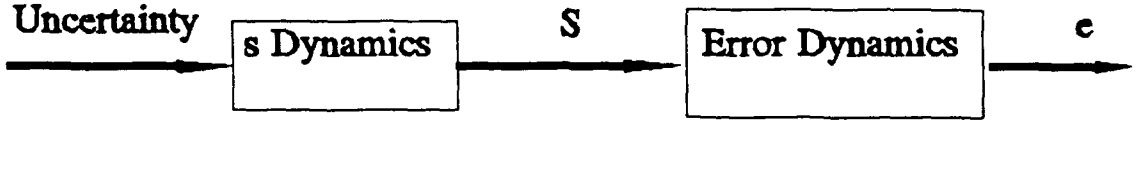

Figure 1 A synthesis view of sliding mode control

II. PLANT AND ITS MATHEMATICAL MODELS

A. PHYSICAL PLANT

An experimental arm is shown in Fig. 2, which is driven by an electrohydraulic actuator. The motion of the arm is limited to a vertical plane. The flexible arm can bend freely in the vertical plane, but it is stiff in torsion and horizontal bending. The arm consists of two parallel steel flat bars welded at the base and directly clamped to the hydraulic actuator. Torsional stiffness is obtained by connecting the two steel bars, with thin steel strips, to seven transverse steel bridges. Table 1 shows geometric and mass properties of the flexible arm.

Table 1

Arm Length	0.9985 m
Arm Mass	4.8565 kg
Transverse Rigidity	81.3 N m^2
Arm Cross-Sectional Area	6.178×10^{-4} m^2
Density	7861.05 kg/m^3

A potentiometer is used to measure the actuator (large motion) signal. A two-arm bridge strain gage attached to the center of the arm is used to calculate the tip deflection (small motion) of the arm. Data acquisition was performed using a high speed Data Translation interface board

DT 2821-F-8DI, which was installed in a micro-computer (standard IBM-PC AT). The support software (AT-LAB) allowed direct manipulation of the data acquisition board through the use of provided subroutines which are compatible with FORTRAN.

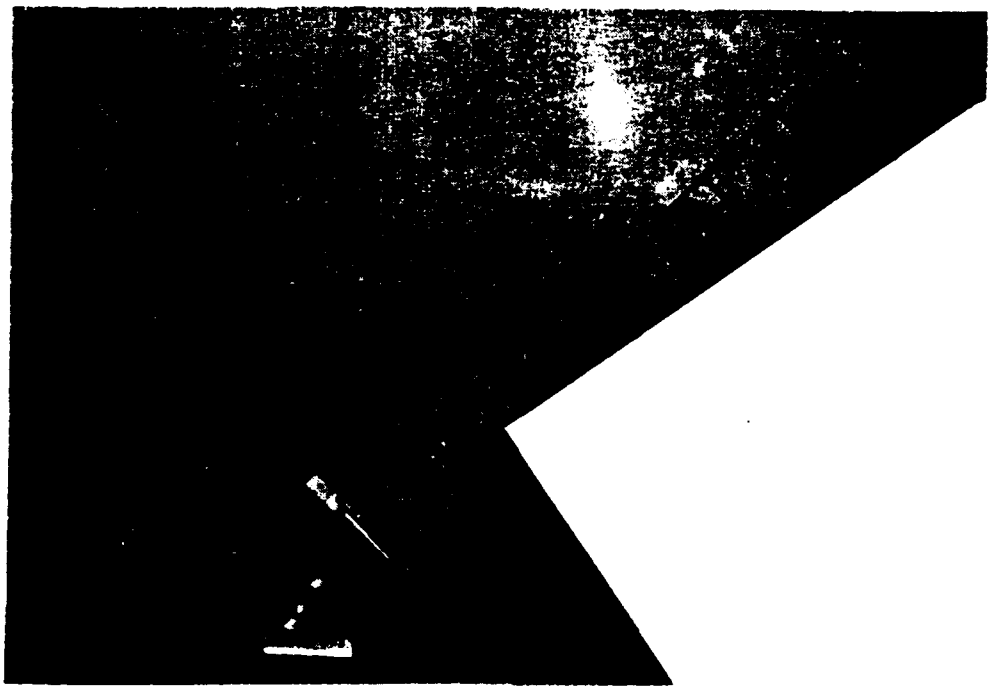

Figure 2 A Single-Link Flexible Manipulator System

B. MATHEMATICAL MODEL OF THE PLANT

In this section, a mathematical model of the plant is given, which includes an ERLS (Equivalent Rigid Link System) dynamic model of the arm and actuator dynamics.

1. AN ERLS DYNAMIC MODEL OF FLEXIBLE ARM

The ERLS is defined as a hypothetical system which produces the large motion and whose kinematics are equivalent to a rigid-link system. An ERLS of a planar manipulator

with single link is shown schematically with the dash line in Figure 3, where geometric center line of link was drawn. The solid lines stand for the deformed state of the arm. The ERLS of a flexible arm describes the large motion of the arm, and then the small motion arising from the structure flexibility can be superimposed on the ERLS [Ref. 1]

The ERLS dynamic model of the arm was developed by means of Lagrange's formulation, the Finite Element Method, and the ERLS kinematics. To apply the Lagrangian dynamics to the flexible arm, the generalized coordinate (Figure 3) is chosen to describe the large motion by joint variable θ. The Finite Element Method is utilized to discretize the displacement such that the small motion is represented in terms of nodal displacement υ, where υ is the tip deflection. In this study, the natural mode shape functions of a beam are used to represent the flexural motion of the flexible arm, and the arm is modeled as a continuous Euler-Bernoulli cantilever beam, neglecting shear deformation and rotary effects.[Ref. 4]

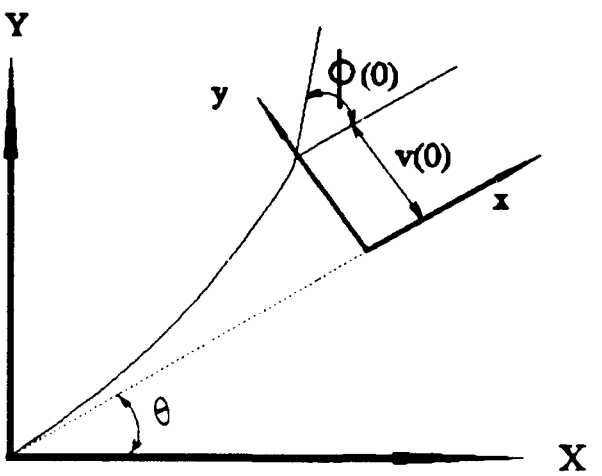

Figure 3 Generalized Coordinate

2. EQUATIONS OF MOTION OF FLEXIBLE ARM

Lagrange's equations for the flexible manipulator are

$$\frac{d}{dt}\left(\frac{\partial KE}{\partial \dot{q}_i}\right) - \frac{\partial KE}{\partial q_i} + \frac{\partial PE}{\partial q_i} = GF_i \quad i=1,2. \tag{2.1}$$

where KE and PE are kinetic and potential energies. q_i's are the generalized coordinates and are defined by

$$[q_1 \quad q_2] = [\theta \quad \upsilon] \tag{2.2}$$

GF's are generalized forces. For the system without applied forces at the end effector, the generalized force vector is given as

$$GF = [GF_1 \; GF_2]^T = [T \; 0]^T \tag{2.3}$$

where T is an applied torque at the joint. $[\ ]^T$ represents the transpose of the matrix.

The total kinetic energy has three parts, i.e., the kinetics energy of the arm (KE_l), the rotor of the actuator (KE_r), and the payload (KE_p). The mathematical expressions of these energies are

$$KE_l = \frac{1}{2}\int_{link} \dot{r}_l^T \dot{r}_l \, dm \tag{2.4}$$

$$KE_r = \frac{1}{2} \, Trace \int_{rotor} \dot{r}_r \, \dot{r}_r^T \, dm \tag{2.5}$$

and

$$KE_p = \frac{1}{2} \, Trace \int_{payload} \dot{r}_p \, \dot{r}_p^T \, dm \tag{2.6}$$

where $\dot{r}$ is the velocity of a differential mass, the subscripts l , r, and p are for the link, the rotor, and the payload, and dm is the differential mass. Note that applying the Trace operator on the kinetic improves the computational efficiency for the rigid-body modeling since time-invariant terms can be separated from time-variant terms through the operator.

The potential energy comes from strain energy and gravitational energy. The mathematical expressions are given as

$$PE_s = \frac{1}{2} \int_{link} EI_{zz} \left(\frac{\partial^2 v_l}{\partial x^2}\right)^2 dx \tag{2.7}$$

and

$$PE_g = - \int_{link} r_l^T \, g \, dm - \int_{payload} r_p^T \, g \, dm \tag{2.8}$$

where EI_{zz} is the bending rigidity in the xy plane, g is a gravitational acceleration vector. According to the inertial coordinates in Figure 3, the gravitational acceleration is defined as

$$g = [0\ 0\ -9.80]^T \quad (m/sec^2) \tag{2.9}$$

KE and PE are expressed in terms of the generalized coordinates q_i. The absolute positions and velocities in Equation (2.4)-(2.8) are formulated by the kinematics of the ERLS. [Ref. 1]

Two sets of equation of motion for a single-link flexible manipulator are obtained from the derivation of above, which are nonlinear, coupled, second-order, ordinary differential equations represented as follows,

$$m_{\theta\theta}\ddot{\theta} + m_{\theta\eta}\ddot{\upsilon} = f_\theta \tag{2.10}$$

$$m_{\eta\theta}\ddot{\theta} + m_{\eta\eta}\ddot{\upsilon} + g_\eta\dot{\upsilon} + k_\eta\upsilon = f_\eta \tag{2.11}$$

where υ represents the nodal displacement. $m_{\theta\theta}$ and $m_{\eta\eta}$ are effective masses for the large motion and small motion, $m_{\theta\eta}$ and $m_{\eta\theta}$ are coupled masses between large and small motion, g_η is gyroscope for small motion, k_η is stiffness for small motion, and f_θ and f_η are load for large and small motion. The effective and coupled masses, the gyroscopic matrix, and load are nonlinear in θ or $\dot{\theta}$. In order to separate the applied torque from other terms in f_θ, let $f_\theta = h_\theta + T$. [Ref. 3] In this study, a motion control is designed to control the tip position of the arm. With a small deflection assumption, the tip position can be approximated and represented by a total angle φ, where $\varphi = \theta + \upsilon/L$. The control design therefore requires

a mathematical model for the total angle. Equations (2.10) and (2.11) can thus be rewritten into an explicit form for the applied torque as follows,

$$m_{\theta\theta}\ddot{\theta} + m_{\theta\eta}\ddot{\upsilon} = h_{\theta} + T \tag{2.12}$$

$$m_{\eta\theta}\ddot{\theta} + m_{\eta\eta}\ddot{\upsilon} + g_{\eta}\dot{\upsilon} + k_{\eta}\upsilon = f_{\eta} \tag{2.13}$$

Since

$$\ddot{\theta} = \ddot{\varphi} - \frac{\ddot{\upsilon}}{L}$$

Equations (2.12) and (2.13) can further be rearranged in terms of the total angle as

$$m_{\theta\theta}\ddot{\varphi} + (m_{\theta\eta} - m_{\theta\theta}/L)\ddot{\upsilon} = h_{\theta} + T \tag{2.14}$$

$$m_{\eta\theta}\ddot{\varphi} + (m_{\eta\eta} - m_{\eta\theta}/L)\ddot{\upsilon} + g_{\eta}\dot{\upsilon} + k_{\eta}\upsilon = f_{\eta} \tag{2.15}$$

From Equations (2.14) and (2.15), an uncoupled equation for the total angle φ is obtained by eliminating the small deflection υ as follows,

$$N\ddot{\varphi} + F_{c} = T \tag{2.16}$$

where

$$N = m_{\theta\theta} - D\, m_{\eta\theta}$$

$$F_c = D\,(f_\eta - g_\eta\,\dot{\upsilon} - k_\eta\,\upsilon) - h_\theta \tag{2.17}$$

$$D = \frac{m_{\theta\eta} - m_{\theta\theta}/L}{m_{\eta\eta} - m_{\eta\theta}/L}$$

3. ELECTROHYDRAULIC ACTUATION

The flexible arm is driven by an electrohydraulic actuator of which the dynamics is an integral part of the total system model. The dynamics of the electrohydraulic actuator include servovalve dynamics and hydraulic motor dynamics. A simplified description of servovalve dynamics was provided by MOOG, the manufacturer of the servovalve. A single equation which presents the dynamics is given as

$$Q = K\, I \sqrt{P_v} \tag{2.18}$$

where Q is the flow delivered from the servovalve, K is a valve sizing constant which contributes to hydraulic system damping. I is an input current, and P_v is the valve pressure drop, i.e., $P_s - P_L$, where P_s is the supply pressure and P_L is the load pressure drop.

Motor dynamics consists of a form of continuity equation and the torque output equation [Ref. 13]. They are written as follows

$$Q = D_m\dot{\theta} + C_{tm}P_L + \frac{V_t}{4\beta_e}\dot{P}_L \tag{2.19}$$

and

$$T = \eta_t P_L D_m \tag{2.20}$$

where D_m is the motor displacement, $C_{tm}P_L$ is the leakage flow in the motor, $(V_t / 4\beta_e)\dot{P}_L$ is the compressibility flow, and η_t is the torque efficiency. A detail account for the selection of hydraulic component for the system was included in [Ref. 3].

C. STATE SPACE REPRESENTATION

Considering the plant dynamic model, Equations (2.16) through (2.20), these equations can further be rearranged as

$$N\bar{\varphi} + F_c = \eta_t D_m P_L \tag{2.21}$$

and

$$D_m\dot{\theta} + C_{tm}P_L + (\frac{V_t}{4\beta_e})\dot{P}_L = KI\sqrt{P_v} \tag{2.22}$$

Differentiating Equation (2.21)

$$N\bar{\varphi} + \dot{N}\ddot{\varphi} + \dot{F}_c = \eta_t D_m \dot{P}_L \tag{2.23}$$

Equation (2.21) and (2.22) can be rewritten as

$$\dot{P}_L = \left(\frac{4\beta_e}{V_t}\right)\left(KI\sqrt{P_v} - D_m\dot{\theta} - C_{tm}P_L\right) \tag{2.24}$$

$$P_L = \left(\frac{1}{\eta_t D_m}\right)\left(N\ddot{\varphi} + F_c\right) \tag{2.25}$$

Substituting Equations (2.25) into Equation (2.24), and then substituting Equation (2.24) into Equation (2.23) give

$$\dddot{\varphi} = b_0 I + f_0 \tag{2.26}$$

where

$$b_0 = \frac{4\beta_e\eta_t D_m K}{V_t N}\sqrt{P_s - \frac{N\ddot{\varphi} + F_c}{\eta_t D_m}}$$

$$f_0 = -\left(\frac{4\beta_e C_{tm}F_c}{V_t N} + \frac{\dot{F}_c}{N} + \frac{4\beta_e\eta_t D_m^{\,2}}{V_t N}\dot{\theta}\right) - \left(\frac{\dot{N}}{N} + \frac{4\beta_e C_{tm}}{V_t}\right)\ddot{\varphi}$$

Note that Equation (2.26) is a time-varying, nonlinear third-order ordinary differential equation represented in scalar form.

In this study, the state space representation of Equation (2.26) will be used for the control action and the representation is given as follows

$$\dot{\boldsymbol{X}} = \boldsymbol{B}\boldsymbol{u} + \boldsymbol{f} \tag{2.27}$$

where

$$\boldsymbol{X} = [\ \varphi \quad \dot{\varphi} \quad \ddot{\varphi}\ ]^T$$

$$\boldsymbol{B} = [\ 0 \quad 0 \quad b_0]^T$$

$$\boldsymbol{u} = I$$

$$\boldsymbol{f} = [\ \dot{\varphi} \quad \ddot{\varphi} \quad f_0\ ]^T$$

The representation of Equation (2.27) will be used for the development in the control techniques.

III. CONTROLLER DESIGN (SLIDING MODE CONTROL)

A. INTRODUCTION

For many control system design problems, the designers do not have a detailed state-space model of the plant to be controlled, either because it is too complex, or because its dynamics are not completely understood. No nominal model should be considered complete without an assessment of its errors. These errors are refereed to as *modeling uncertainties* of the system. Therefore, a robust control design was attempted such that the control system will be insensitive to the modeling uncertainties.

Based on our knowledge of physical mechanisms which cause differences between model and plant and our ability of representing these mechanisms, the representations of uncertainties may vary in terms of the structure they have. In practice, it is possible to represent these error in a highly structured parameterized form. These are usually the low frequency components. For a manipulator system, these parametric uncertainty or structured uncertainty may come from the imprecision on the manipulator mass properties, unknown loads, uncertainty on the load position in the end-effector, and inaccuracy on the torque constants of the actuators. However, there are always remaining higher frequency errors in the systems, which can not be covered in the parameter uncertainties. Also, these high frequency unmodeled dynamics is referred to as unstructured uncertainties. These unstructured uncertainties are usually caused by unmodeled structural modes, neglected time-delays in the actuators, or finite sampling rate.

The greatest challenge of designing a robust controller is not only to minimize

performance sensitivity to uncertainties due to system parameters but also not to excite the high frequency unmodeled dynamics. The Sliding Mode Control has been known as a robust control that is able to work under the confined uncertainties of dynamic modeling error and environmental disturbances. The concepts of the SMC derived from the Variable Structure Control have been extensively studied in the Soviet Union for more than two decades. The SMC utilizes a high-speed switching control law to drive the plant's state trajectory toward a specified surface (the sliding surface) and to maintain the plant's state trajectory on this surface.

The SMC using output models involves two filters, i.e., S dynamics and error dynamics. The S dynamics was designed to filter the uncertainties and the error dynamics was designed to obtain system error e. Figure 1 illustrated the sliding algorithm from the filter point of view, and also the control system block diagram is shown in Figure 4. Since a high-speed switched control about the sliding surface is not favorable to mechanical systems, a boundary layer thickness was introduced into the SMC such that a smoothed control is achieved.

In this chapter, three sliding mode controls will be presented, which includes the sliding control with a first-order sliding condition, the straight sliding control, and the versatile sliding control. Before presenting the control algorithms, the uncertainties and the sliding surfaces will first be described.

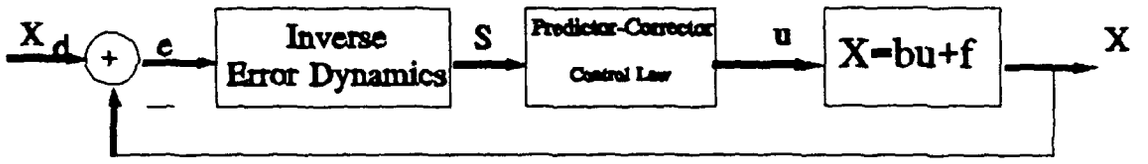

Figure 4 Block Diagram of Sliding Mode Control

B. SYSTEM EQUATIONS AND CONFINED UNCERTAINTIES

From Chapter II, the dynamic equation of a flexible single-link arm was written as

$$\dot{X} = Bu + f \tag{3.1}$$

where

$$X = [x_1 \; x_2 \; x_3]^T$$

$$u = [\, u \,]$$

$$f = [\, f_1 \;\; f_2 \;\; f_3 \,]^T$$

$$B = [\, B_1 \;\; B_2 \;\; B_3 \,]^T$$

Note that B and f are, in general, nonlinear functions of output variables and time. Equation (3.1) can be seen as a canonical form of physical models for nonlinear time-varying dynamical systems.

To control system, a nominal mathematical model of the system can be obtained as,

$$\dot{X} = \hat{B}u + \hat{f} \tag{3.2}$$

$\hat{B}$ and $\hat{f}$ are the nominal values of B and f, which can be estimated from the theoretical evaluations or experiments. Due to parameter uncertainty, the discrepancies between the model and the physical plant are specified by ΔB and Δf, where

$$\begin{aligned} \Delta B &= B - \hat{B} \\ \Delta f &= f - \hat{f} \end{aligned} \tag{3.3}$$

The sliding control assumes that uncertainties and disturbances are bounded. The uncertainties associated with the model are confined as

$$\begin{aligned} |\Delta B| &\le \beta \\ |\Delta f| &\le \gamma \end{aligned} \tag{3.4}$$

where $|\bullet|$ denotes a norm of $\bullet$ which is a vector or a matrix. Note that in this study the norm of $\mathbf{A}$, for instance, is defined as

$$|\mathbf{A}| = \sqrt{[eig(\mathbf{A}^T\mathbf{A})]_{max}} \tag{3.5}$$

where $[eig(\bullet)]_{max}$ stands for the maximum eigenvalue (i.e., the spectral radius) of $\bullet$. This matrix norm is called the spectral norm and the corresponding vector norm is called Euclidean vector norm. The confining parameter β and γ can be found as

$$\beta = \left| \frac{B_{max} - B_{min}}{2} \right| \qquad \gamma = \left| \frac{f_{max} - f_{min}}{2} \right| \tag{3.6}$$

where the components of $\bullet_{max}$ (or $\bullet_{min}$) are the maximum (or minimum) values of the corresponding components of $\bullet$. It is noted that β and γ are non-negative numbers. It also implies that the system parameters and their nominal values are related as

$$\hat{B} = \frac{B_{max} + B_{min}}{2} \qquad \hat{f} = \frac{f_{max} + f_{min}}{2} \tag{3.7}$$

The greatest value of confinement of ΔB is defined as

$$(\Delta B)_{max} = B_{max} - \hat{B} \tag{3.8}$$

C. SLIDING SURFACES

In this single-link flexible arm study, sliding surface will only have one because of one control-input. The sliding surface is defined to be $\boldsymbol{S} = \boldsymbol{0}$ where $\boldsymbol{S}$ is a sliding variable. The sliding variable directly relates the system error as

$$\boldsymbol{S} = \boldsymbol{R}^T \boldsymbol{e} \tag{3.9}$$

where

$$\boldsymbol{S} = [\, s_1 \,]$$

$$\boldsymbol{e} = [\, e_1 \;\; e_2 \;\; e_3 \,]^T$$

$$\boldsymbol{R} = [\, r_1 \;\; r_2 \;\; r_3 \,]^T$$

$\boldsymbol{e}$ is the tracking error, i.e., $\boldsymbol{e} = \boldsymbol{X} - \boldsymbol{X}_d$, where $\boldsymbol{X}_d$ is a desired output vector, and is defined as

$$\boldsymbol{X}_d = [\, x_{1d} \;\; x_{2d} \;\; x_{3d} \,]^T$$

$\boldsymbol{R}$ is to locate the poles of the error dynamics on the sliding surface. $\boldsymbol{S}$ can be interpreted as an input to the error dynamics, and the error is a filtered version of S.

An integral control can be introduced into sliding surface to eliminate steady-state error as follows,

$$S = R^T e + G^T \int e dt \tag{3.10}$$

where

$$G = \begin{bmatrix} g_1 & g_2 & g_3 \end{bmatrix}^T$$

Note that once the poles for each sliding surface were assigned, the R and G are both constant vectors.

D. SLIDING CONTROL WITH A FIRST-ORDER SLIDING CONDITION

The relationship between control input u and S dynamics can then be obtained by differentiating Equation (3.10)

$$\dot{S} = R^T \dot{e} + G^T e \tag{3.11}$$

Since

$$\dot{X} = Bu + f$$
$$\dot{e} = \dot{X} - \dot{X}_d$$

Equation (3 11) can be rewritten as

$$\dot{S} = R^T(Bu + f - \dot{X}_d) + G^T e \tag{3.12}$$

The role of the control input u is to control the S dynamics such that the sliding surfaces can

be reached within a finite time. Once the sliding surface is reached, zero steady-state tracking can thus be achieved on the sliding surfaces provided the error dynamics is stable.

The Lyapunov stability criterion is used to derive a sliding condition which specifies the S dynamics. A Lyapunov function V is picked such that $V = \frac{1}{2} S^T S$. By applying the Lyapunov stability criteria, $\dot{V} \leq 0$, a sliding condition is written as

$$S^T \dot{S} \leq 0 \tag{3.13}$$

The sliding condition assures the attractiveness of the S dynamics toward the sliding surfaces. To be able to adjust the sliding speed $\dot{S}$, Equation (3.13) is rewritten as

$$S^T \dot{S} \leq -\eta |S| \tag{3.14}$$

where η is a non-negative real number. It is worth to note that Equation (3.14) is a first-order sliding condition.

The sliding condition specifies a desired dynamics of S on which S slides toward the sliding surfaces and the steady-state error can be eliminated. To obtain the desired S dynamics, the control input can be designed using predictor-corrector scheme as

$$u = (R^T \hat{B})^{-1}(\hat{u} - k\, sgn(S)) \tag{3.15}$$

where $\hat{u}$, i.e., nominal control input, can be obtained by letting $\dot{S} = 0$, which will give

$$\hat{u} = R^T (\dot{X}_d - \hat{f}) - G^T e \tag{3.16}$$

Also,

$$sgn(S) = [\ sgn(s_1)\ ] = \begin{cases} 1 & s_1 \geq 0 \\ -1 & s_1 < 0 \end{cases}$$

Note that $(R^T\hat{B})$ is a scalar. Also, note that $(R^T\hat{B})^{-1}\hat{u}$ is a predictor and $(R^T\hat{B})^{-1}k\ sgn(S)$ is a corrector. The gain k is a non-negative real number and is determined by using matrix-norm techniques. By substituting Equation (3.12) into Equation (3.14), an inequality will be given as

$$S^T \left[R^T(Bu+f-\dot{X}_d)+G^T e\right] \leq -\eta \|S\| \tag{3.17}$$

The control input (Equation (3.15)) can then be substituted into Equation (3.17), which will give

$$S^T\left[(R^TB)(R^T\hat{B})^{-1}(\hat{u}-k\ sgn(S))+R^T(f-\dot{X}_d)+G^T e\right] \leq -\eta \|S\| \tag{3.18}$$

Also, Equation (3.16) will give

$$R^T\dot{X}_d = \hat{u} + R^T\hat{f} + G^T e$$

Substituting into Equation (3.18) and rearranging give

$$S^T \left\{R^T(f-\hat{f})+\left[(R^TB)(R^T\hat{B})^{-1}-I\right]\hat{u}-(R^TB)(R^T\hat{B})^{-1}k\ sgn(S)\right\} \leq -\eta \|S\| \tag{3.19}$$

The uncertainties described in Equation (3.3) are applied to Equation (3.19) and the sliding

condition is given as

$$S^T \{R^T\Delta f+(R^T\Delta B)(R^T\hat{B})^{-1}\hat{u}-[I+(R^T\Delta B)(R^T\hat{B})^{-1}]k\ sgn(S)\}\leq -\eta\|S| \tag{3.20}$$

In order to quantify gain k, the matrix-norm technique is applied to Equation (3.20) and

$$\begin{aligned} &S^T\{R^T\Delta f+(R^T\Delta B)(R^T\hat{B})^{-1}\hat{u}-[I+(R^T\Delta B)(R^T\hat{B})^{-1}]k\ sgn(S)\} \\ &\leq |S|\{|R\|(|\Delta f|+|\Delta B\|(R^T\hat{B})^{-1}\hat{u}|)-k(1-|R\Delta B\|(R^T\hat{B})^{-1}sgn(S)\|)\} \end{aligned} \tag{3.21}$$

Note that

$$-kS^T(R^T\Delta B)(R^T\hat{B})^{-1}sgn(S)\ \leq\ k|S||R\Delta B||(R^T\hat{B})^{-1}sgn(S)|$$

$$-kS^Tsgn(S)\ \leq\ -k|S|$$

and

$$|R^T\Delta B|\ =\ |R^T(\Delta B)_{max}|$$

Therefore,

$$|S|\{|R|(|\Delta f|+|\Delta B||(R^T\hat{B})^{-1}\hat{u}|)-k(1-|R\Delta B||(R^T\hat{B})^{-1}sgn(S)|\}\leq -\eta|S| \tag{3.22}$$

Because of the uncertainty confinements (Equations (3.4) and (3.6)), Equation (3.22) becomes

$$\|S\|\{\|R\|(\gamma+\beta\|(R^T\hat{B})^{-1}\hat{u}\|)-k(1-\|R\Delta B\|\|(R^T\hat{B})^{-1}sgn(S)\|)\}\leq-\eta\|S\| \tag{3.23}$$

Finally, the gain k is found as

$$k\geq\frac{\|R\|(\gamma+\beta\|(R^T\hat{B})^{-1}\hat{u}\|)+\eta}{1-\|R\Delta B\|\|(R^T\hat{B})^{-1}sgn(S)\|} \tag{3.24}$$

As stated before, the gain k is a non-negative number. Thus, a sufficient condition is given

$$1 > \|R\Delta B\|\|(R^T\hat{B})^{-1}sgn(S)\| \tag{3.25}$$

The Lyapunov stability is guaranteed as long as the gain k is chosen according to Equation (3.24). For control design purposes, the minimum value of k is selected since the least control effort is desired. Because of the discontinuity of the *sgn* functions, the control is called switched control and causes chattering, which is unfavorable to the mechanical devices. Equation (3.15) can be substituted into Equation (3.12) to obtain the S dynamics as follows

$$\dot{S}+k(R^TB)(R^T\hat{B})^{-1}sgn(S)=R^T(f-\hat{f})-(R^T\Delta B)(R^T\hat{B})^{-1}\hat{u} \tag{3.26}$$

It is observed that the right-hand-side of Equation (3.26) consists of uncertainties, error, and desired trajectory dynamics, which are the excitations to the S dynamics and are to be filtered out by the first-order S dynamics.

To smooth the control law, a boundary layer with thickness ϕ is introduced where the

thickness is a real positive number. Outside the boundary, $|s_1| \geq \phi$, the control law u is designed to satisfy the sliding condition, Equation (3.14), which guarantees boundary layer attractiveness. Inside the boundary, $|s_1| < \phi$, the control law will impose a smoothing process to the S dynamics. The general law is written as

$$u = (R^T\hat{B})^{-1}\left(\hat{u} - k\ sat(\frac{S}{\phi})\right) \tag{3.27}$$

where

$$\hat{u} = R^T(\dot{X}_d - \hat{f}) - G^Te$$

The *sat* function is defined as

$$sat(\frac{S}{\phi}) = \left[sat(\frac{s_1}{\phi})\right] = \begin{cases} sgn(\frac{s_1}{\phi}) & |s_1| \geq \phi \\ \frac{s_1}{\phi} & |s_1| < \phi \end{cases}$$

The S dynamics of Equation (3.26) is smoothed within the boundary layer, i.e.,

$$\dot{S} + \frac{k}{\phi}(R^TB)(R^T\hat{B})^{-1}S = R^T(\dot{f} - \dot{\hat{f}}) - (R^T\Delta B)(R^T\hat{B})^{-1}\hat{u} \tag{3.28}$$

Equation (3.28) represents a first-order low-pass filter. The boundary layer thickness ϕ determines the response speed of the S dynamics and the values of ϕ can be either constant or

time-varying.

To obtain a time-varying thickness, a maximum bandwidth λ_{max} is assigned to the S dynamics within the boundary layer (Equation (3.28)). The analytic properties of matrix-norm are to be used to quantify the bandwidth. The spectral radius of the bandwidth is defined as

$$\lambda = \frac{k}{\phi}(R^T B)(R^T \hat{B})^{-1}$$

Thus, the boundary layer thickness ϕ is designed such that $\lambda \leq \lambda_{max}$. Also,

$$\frac{k}{\phi}(R^T B)(R^T \hat{B})^{-1} \leq \frac{k}{\phi} \left|(R^T B)(R^T \hat{B})^{-1}\right| \tag{3.29}$$

Therefore,

$$\lambda \leq \frac{k}{\phi} \left|(R^T B)(R^T \hat{B})^{-1}\right| \tag{3.30}$$

Since

$$\left|(R^T B)(R^T \hat{B})^{-1}\right| \leq 1 + \|R\Delta B\| \left|(R^T \hat{B})^{-1}\right| \tag{3.31}$$

and by the confinements of uncertainty (Equation (3.3)), the spectral radius is then found as

$$\lambda \leq \frac{k}{\phi}(1 + \|R\Delta B\| \left|(R^T \hat{B})^{-1}\right|)$$

Hence, the maximum bandwidth can be assigned as

$$\lambda_{max} = \frac{k}{\phi}(1 + \|R\Delta B\|\|(R^T\hat{B})^{-1}\|)$$

Therefore, the thickness ϕ is

$$\phi = \frac{k}{\lambda_{max}}(1 + \|R\Delta B\|\|(R^T\hat{B})^{-1}\|) \qquad (3.32)$$

In the Equation (3.32), the maximum bandwidth λ_{max} determines the boundary layer thickness ϕ which controls the tracking accuracy and response speed. By selecting a proper value of λ_{max}, Equation (3.32) can assure that bandwidth will never exceed λ_{max}.

In summary, the sliding control with first-order sliding condition is designed to have both *S* dynamics and error dynamics low-pass filters where an integral error can be added on error dynamics to eliminate the steady-state error. The *S* dynamics is a first-order low-pass filter, in which the uncertainties are filtered. A detailed synthesis view of *S* dynamics of sliding control with a first-order sliding condition is shown in Figure 5. The continuous control of Equation (3.26) can be used to replace the switching control and eliminates the chattering. A detailed block diagram of sliding control system is illustrated in Figure 6.

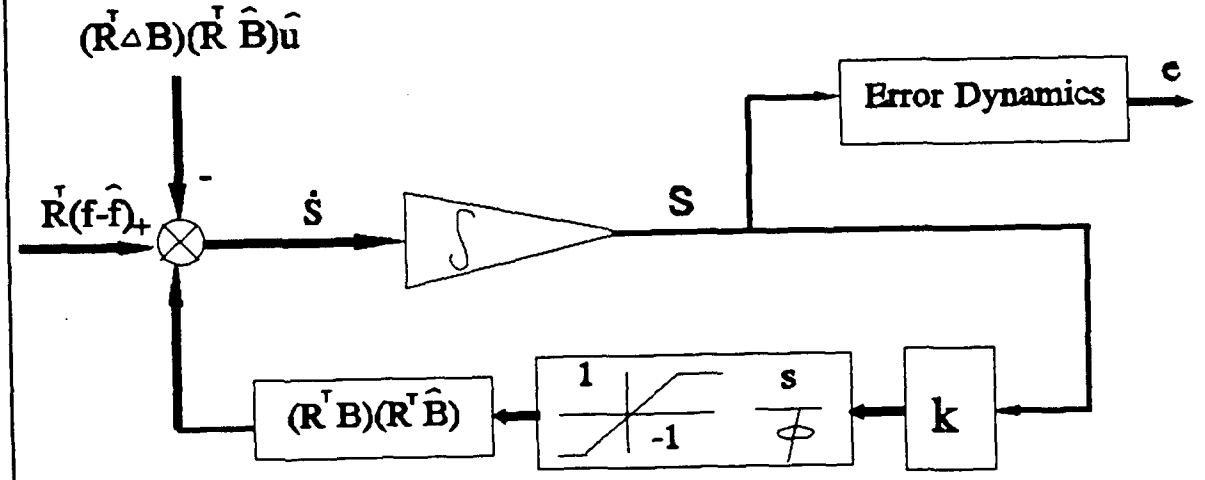

Figure 5 A Detailed Synthesis View of Sliding Mode Control with First-Order Sliding Condition

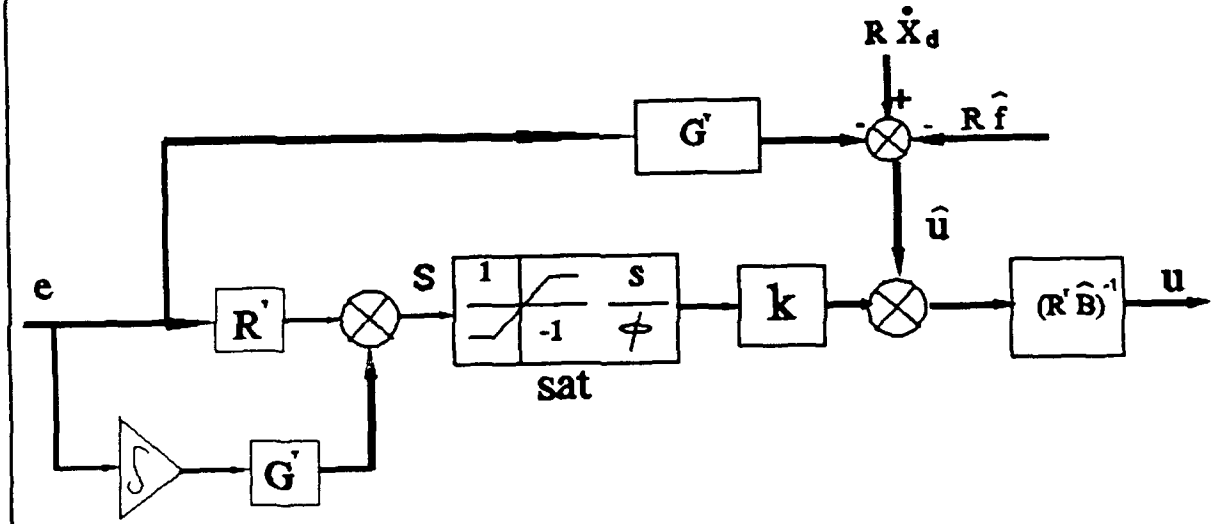

Figure 6 A Detailed Block Diagram of Sliding Mode Control with First-Order Sliding Condition

E. STRAIGHT SLIDING CONTROL WITH FIRST-ORDER PLUS INTEGRAL SLIDING CONDITION

The straight sliding control begins with the same sliding surface which was defined in Equation (3.9). By differentiating Equation (3.9), the relationship between control input and S dynamics can be found

$$\dot{S} = R^T(Bu + f - \dot{X}_d) \tag{3.33}$$

The Lyapunov stability criterion will be used to derive straight sliding condition and specify the desired S dynamics such that the sliding condition can be reached in a finite time. A Lyapunov function is defined as

$$V=\frac{1}{2}S^TS+\frac{1}{2}\left[\int_0^t S^T dt\right]\ \omega_n^2\ \left[\int_0^t S dt\right] \tag{3.34}$$

By applying the Lyapunov stability criterion, $\dot{V} \leq 0$, the sliding condition is obtained

$$S^T(\dot{S}+\omega_n^2\int_0^t S dt) \leq 0 \tag{3.35}$$

Equation (3.35) indicates that a first-order plus integral sliding condition representing a stability criterion for an equivalent mass-spring-damper system in which $\int_0^t S\ dt$ is a equivalent displacement. The $\omega_n^2\int_0^t S\ dt$ provides an additional restoring effort and the dynamic behavior

can thus be tuned by the equivalent spring constant ω_n^2 . Furthermore, Equation (3.35) can be written as

$$S^T(\dot{S}+\omega_n^2\int_0^t S\ dt)\leq -\eta\|S\| \tag{3.36}$$

where η is a positive real number. The sliding condition, Equation (3.36), describes a desired S dynamics on which S slides toward the sliding surface.

In order to obtain the desired S dynamics, the control input can be obtained as

$$u = (R^T\hat{B})^{-1}(\hat{u} - ksgn(S)) \tag{3.37}$$

where $\hat{u}$ is obtained by letting $\dot{S}+\omega_n^2\int Sdt=0$,

$$\hat{u} = R^T\hat{B}\tilde{u} = R^T(\dot{X}_d-\hat{f})-\omega_n^2\int Sdt \tag{3.38}$$

The gain k is determined by substituting Equation (3.33) into Equation (3.36), which gives

$$S^T\left(R^T(Bu+f-\dot{X}_d)+\omega_n^2\int Sdt\right)\leq -\eta\|S\| \tag{3.39}$$

Equation (3.39) can be further simplified by using Equation (3.37) and Equation (3.38),

$$S^T\{R^T\Delta f+(R^T\Delta B)(R^T\hat{B})^{-1}\hat{u}-[I+(R^T\Delta B)(R^T\hat{B})^{-1}]k\ sgn(S)\}\leq -\eta\|S\| \tag{3.40}$$

It is noted that Equation (3.40) has exactly the same form as Equation (3.20) has. Thus, the same matrix-norm techniques used in Equations (3.20-25) will be utilized to quantify the gain k, which gives

$$k \geq \frac{\|R\|\left(\gamma + \beta \|(R^T\hat{B})^{-1}\hat{u}\|\right)}{1 - \|R\Delta B\| \|(R^T\hat{B})^{-1} sgn(S)\|} \tag{3.41}$$

The S dynamics can be formed in terms of uncertainties by substituting Equations (3.37-38) into Equation (3.33), which gives

$$\begin{aligned} &\dot{S} + k(R^TB)(R^T\hat{B})^{-1} sgn(S) + (R^TB)(R^T\hat{B})^{-1}\omega_n^2 \int S dt \\ &= \left[(R^TB)(R^T\hat{B})^{-1} - I\right] R^T\dot{X}_d + R^Tf - (R^TB)(R^T\hat{B})^{-1}R^T\hat{f} \end{aligned} \tag{3.42}$$

The right-hand-side of Equation (3.42) represents the excitations to the S dynamics, which consist of the desired trajectory dynamics, uncertainties, and disturbances. In other words, the S dynamics can be treated as a filtered version of excitations and filter out most unwanted uncertainties and disturbances. The *sgn* function causes chattering which will produce undesirable noise and mechanical wear. A continuous control is then developed to eliminated chattering.

A boundary layer with thickness ϕ is introduced to smooth out S dynamics. The control algorithm is defined as

$$u = (R^T\hat{B})^{-1}\left(\hat{u} - k\, sat(\frac{S}{\phi})\right)$$

where

$$\hat{u} = R^T(\dot{X}_d - \hat{f}) - \omega_n^2 \int_0^t S \, dt$$

$$sat(\frac{S}{\phi}) = \left[sat(\frac{s_1}{\phi}) \right] = \begin{cases} sgn(\frac{s_1}{\phi}) & |s_1| \geq \phi \\ \frac{s_1}{\phi} & |s_1| < \phi \end{cases}$$

Thus, a smoothed first-order low-pass S dynamics can be obtained as

$$\begin{aligned} &\dot{S} + \frac{k}{\phi}(R^TB)(R^T\hat{B})^{-1}S + (R^TB)(R^T\hat{B})^{-1}\omega_n^2 \int S dt \\ &= \left[(R^TB)(R^T\hat{B})^{-1} - I\right]R^T\dot{X}_d + R^Tf - (R^TB)(R^T\hat{B})^{-1}R^T\hat{f} \end{aligned} \tag{3.43}$$

The S integral introduces an integral control to the S dynamics and guarantees a zero steady-state values of S which will drive the steady-state error dynamics to zero.

From Equation (3.43), the maximum bandwidth λ_{max} is assigned to S dynamics within the boundary layer. The analytic properties of matrix-norm are to be used to quantify the bandwidth. The bandwidth (λ) is defined as

$$\lambda^2 = (R^TB)(R^T\hat{B})^{-1}\omega_n^2$$

also,

$$(R^TB)(R^T\hat{B})^{-1}\omega_n^2 \leq \left\|(R^TB)(R^T\hat{B})^{-1}\right\|\omega_n^2 \tag{3.44}$$

Therefore,

$$\lambda^2 \leq \left|(R^T B)(R^T \hat{B})^{-1}\right| \omega_n^2 \tag{3.45}$$

Since,

$$\left|(R^T B)(R^T \hat{B})^{-1}\right| \leq 1 + \|R\Delta B\| \left|(R^T \hat{B})^{-1}\right| \tag{3.46}$$

The bandwidth is then found as

$$\lambda^2 \leq (1 + \|R\Delta B\| \left|(R^T \hat{B})^{-1}\right|) \omega_n^2 \tag{3.47}$$

Hence, the maximum bandwidth is

$$\lambda_{max}^2 = (1 + \|R\Delta B\| \left|(R^T \hat{B})^{-1}\right|) \omega_n^2 \tag{3.48}$$

Since the maximum bandwidth has been chosen such that the bandwidth of the S dynamics will never exceeds λ_{max} and the unmodeled dynamics can be filtered. From the damping of Equation (3.43) and also $B = \Delta B + \hat{B}$

$$\frac{k}{\phi}(R^T B)(R^T \hat{B})^{-1} = \frac{k}{\phi}\left[(R^T \Delta B)(R^T \hat{B}) + 1\right]$$

Also,

$$\frac{k}{\phi}\left[(R^T \Delta B)(R^T \hat{B})^{-1} + 1\right] \geq \frac{k}{\phi}(R^T \Delta B)(R^T \hat{B})^{-1} \tag{3.49}$$

Therefore, a lower bound of damping is

$$2\zeta_l\lambda_l = \frac{k}{\phi}(\boldsymbol{R}^T\Delta\boldsymbol{B})(\boldsymbol{R}^T\hat{\boldsymbol{B}})^{-1} \tag{3.50}$$

Note that $(\boldsymbol{R}^T\Delta\boldsymbol{B})$ and $(\boldsymbol{R}^T\hat{\boldsymbol{B}})$ are both scalar. Thus,

$$\zeta_l = \left(\frac{k}{2\lambda_l\phi}\right)(\boldsymbol{R}^T\Delta\boldsymbol{B})(\boldsymbol{R}^T\hat{\boldsymbol{B}})^{-1} \tag{3.51}$$

λ_l is replaced by λ_{max}, which gives

$$\zeta_l \geq \left(\frac{k}{2\lambda_{max}\phi}\right)(\boldsymbol{R}\Delta\boldsymbol{B})(\boldsymbol{R}^T\hat{\boldsymbol{B}})^{-1} \tag{3.52}$$

Hence, a more conservative lower bound of damping is obtained as

$$\zeta_L = \left(\frac{k}{2\lambda_{max}\phi}\right)(\boldsymbol{R}\Delta\boldsymbol{B})(\boldsymbol{R}^T\hat{\boldsymbol{B}})^{-1}$$

Therefore, the thickness ϕ is,

$$\phi = \left(\frac{k}{2\zeta_L\lambda_{max}}\right)(\boldsymbol{R}\Delta\boldsymbol{B})(\boldsymbol{R}^T\hat{\boldsymbol{B}})^{-1} \tag{3.53}$$

λ_{max} is given by Equation (3.48), where the spring constant ω_n^2 is the only control parameter for the bandwidth. Once λ_{max} is determined, another tuning parameter ζ_L can be properly adjusted. It is shown in Equation (3.53) that ζ_L also governs the activeness of S dynamics where the activeness refers to the magnitude level of S response. The greater ζ_L is, the thinner the

boundary layer thickness ϕ is, and the less active of S dynamics is. Also, the less active S dynamics has less influence over to the error dynamics. However, the thinner boundary layer may require higher sampling ratio to eliminate the chattering.

In summary, the straight sliding control is designed to have two lower-pass filters in which S dynamics is designed as a first-order low-pass filter with an integral of S . Figure 7 shows a detailed synthesis view of straight sliding control system. The integral of $\mathbb{S}$ provides an ability to make the steady-state value of S to zero, and the steady-state error will be driven to zero. The straight sliding control provides two independent tuning parameters to adjust the S dynamics. The tuning parameter of spring constant (ω_n^2) provides a restriction to the bandwidth of S dynamics which the unstructured uncertainties (unmodeled dynamics) will be rejected. In addition, the ζ_L gives a fine tune on the thickness of boundary layer. Figure 8 show a detailed block diagram of straight sliding control system.

F. VERSATILE SLIDING CONTROL WITH SECOND-ORDER SLIDING CONDITION

The idea of versatile sliding control is to use a low-pass S dynamics filter to filter out unwanted high-frequency noises that include unmodeled dynamics and uncertainties, and then followed by a high-pass (or band-pass) error dynamics filter.

In order to make error dynamics a high-pass filter, an additional zero is placed into the error dynamics. Thus, the error dynamics is modified to be

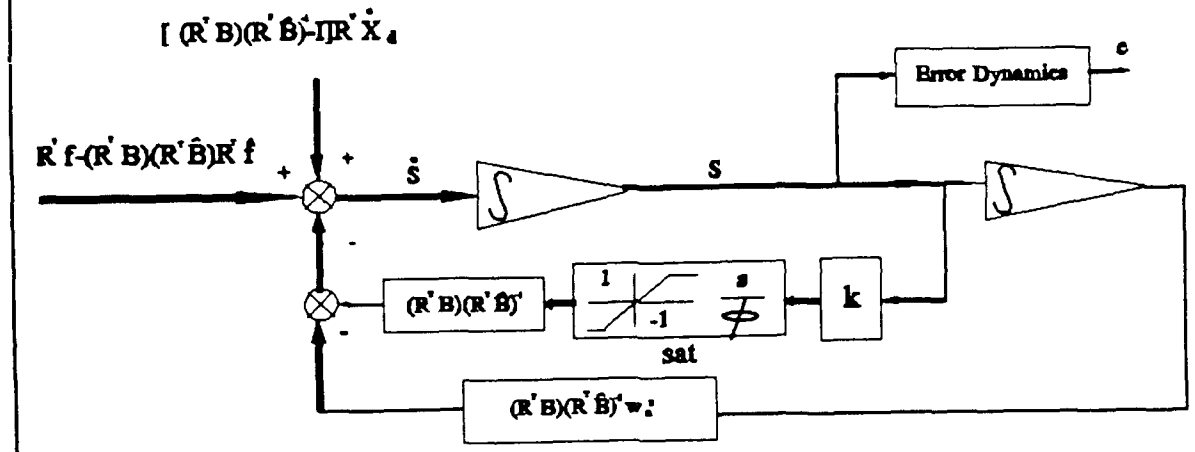

Figure 7 A Detailed Synthesis View of Straight SLiding Control

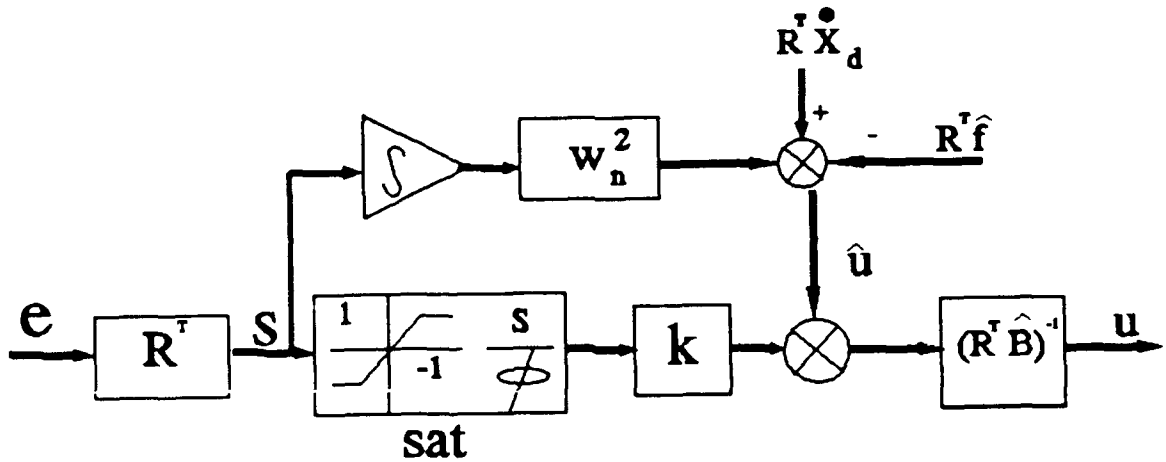

Figure 8 A Detailed Block Diagram of Straight Sliding Control

$$\dot{S}+zS=R^T e+G^T\int e dt \tag{3.54}$$

z is a scalar where a zero is assigned for sliding surface. The relationship between input control and S dynamics can be found by differentiating Equation (3.54), which gives

$$\ddot{S}+z\dot{S}=R^T(Bu+f-\ddot{X}_d)+G^T e \tag{3.55}$$

Now, Equation (3.55) is a second-order form of sliding condition, which specifies S dynamics in order to reach sliding surface. Thus a Lyapunov function to suit the second-order sliding condition is

$$V = \frac{1}{2}\dot{S}^T\dot{S}+\frac{1}{2}S^T\omega_n^2 S \tag{3.56}$$

By differentiating Equation (3.56) and imposing sliding speed parameter, the Lyapunov stability criteria becomes

$$\dot{S}^T(\ddot{S}+\omega_n^2 S)\leq -\eta|\dot{S}| \tag{3.57}$$

where sliding speed, η, is a positive real number.

According to the second-order sliding condition, the switched control law will be

$$u = (R^T\hat{B})^{-1}(\hat{u} - k\ sgn(\dot{S})) \tag{3.58}$$

where the nominal $\hat{u}$ can be obtained by letting $\ddot{S} + \omega_n^2 S = 0$

$$\hat{u}=R^T(\ddot{X}_d-\hat{f})-G^Te+z\dot{S}-\omega_n^2S \tag{3.59}$$

and the gain k can be found based on the same knowledge as before. Substituting Equation (3.55) into Equation (3.57) and further simplifying by Equation (3.58) and Equation (3.59) give,

$$\dot{S}^T\{R^T\Delta f+(R^T\Delta B)(R^T\hat{B})^{-1}\hat{u}-[1+(R^T\Delta B)(R^T\hat{B})^{-1}]k\ sgn(\dot{S})\}\le-\eta\|\dot{S}\| \tag{3.60}$$

Applying the matrix-norm technique, the gain k can be found as

$$k\ge\frac{\|R\|(\gamma+\beta\|(R^T\hat{B})^{-1}\hat{u}\|)+\eta}{1-\|R\Delta B\|\|(R^T\hat{B})^{-1}sgn(\dot{S})\|} \tag{3.61}$$

Because of chattering, the switched control law is modified to be a continuous control law as,

$$u\ =\ (R^T\hat{B})^{-1}\left(\hat{u}\ -\ k\ sat(\frac{\dot{S}}{\phi})\right) \tag{3.62}$$

Outside the boundary layer, the control law is governed by Equation (3.58). While inside the boundary layer, Equation (3.62) governs the control law. Thus, the definition of *sat* function is defined as

$$sat(\frac{\dot{S}}{\phi})\ =\ \left[\ sat(\frac{\dot{s}_1}{\phi})\ \right]\ =\ \begin{cases} sgn(\frac{\dot{s}_1}{\phi}) & |s_1|\ge\phi \\ \frac{\dot{s}_1}{\phi} & |s_1|<\phi \end{cases}$$

The continuous S dynamics can then be obtained by substituting Equation (3.59) and Equation (3.62) into Equation (3.55)

$$\begin{aligned}&\ddot{S}+\left[\frac{k}{\phi}(R^TB)(R^T\hat{B})^{-1}+\left(I-(R^TB)(R^T\hat{B})^{-1}\right)z\right]\dot{S}+(R^TB)(R^TB)^{-1}\omega_n^2S\\&=R^Tf-(R^TB)(R^T\hat{B})^{-1}R^T\hat{f}+\left[(R^TB)(R^T\hat{B})^{-1}-I\right]R^T\dot{X}_d+\left[I-(R^TB)(R^T\hat{B})^{-1}\right]G^Te\end{aligned} \tag{3.63}$$

To find time-varying boundary layer thickness, the maximum bandwidth is first to be evaluated as in the section D (straight sliding control),

$$\lambda_{max}^2 = (1+\|R\Delta B\|\|(R^T\hat{B})^{-1}\|)\omega_n^2 \tag{3.64}$$

The damping of S dynamics is

$$\begin{aligned}&\frac{k}{\phi}(R^TB)(R^T\hat{B})^{-1}+\left(I-(R^TB)(R^T\hat{B})^{-1}\right)z\\&=\frac{k}{\phi}\left((R^T\Delta B)(R^T\hat{B})^{-1}+1\right)-(R^T\Delta B)(R^T\hat{B})^{-1}z\end{aligned} \tag{3.65}$$

Also,

$$\begin{aligned}&\frac{k}{\phi}\left((R^T\Delta B)(R^T\hat{B})^{-1}+1\right)-(R^T\Delta B)(R^T\hat{B})^{-1}z\\&\geq\frac{k}{\phi}(R^T\Delta B)(R^T\hat{B})^{-1}-(R^T\Delta B)(R^T\hat{B})^{-1}z\end{aligned} \tag{3.66}$$

Therefore, a lower bound of damping is obtained

$$2\zeta_l\lambda_l = \frac{k}{\phi}(R^T\Delta B)(R^T\hat{B})^{-1} - (R^T\Delta B)(R^T\hat{B})^{-1}z \tag{3.67}$$

and

$$\zeta_l = \left(\frac{(R^T\Delta B)(R^T B)^{-1}}{2\lambda_l}\right)\left(\frac{k}{\phi} - z\right) \tag{3.68}$$

λ_l is replaced by λ_{max}, which gives

$$\zeta_l \geq \left(\frac{(R^T\Delta B)(R^T B)^{-1}}{2\lambda_{max}}\right)\left(\frac{k}{\phi} - z\right) \tag{3.69}$$

Thus, a more conservative lower bound of damping is obtained as

$$\zeta_L = \left(\frac{(R^T\Delta B)(R^T B)^{-1}}{2\lambda_{max}}\right)\left(\frac{k}{\phi} - z\right)$$

Therefore, a time-varying boundary layer thickness is,

$$\phi = \frac{k(R^T\Delta B)(R^T B)^{-1}}{2\zeta_L\lambda_{max} + (R^T\Delta B)(R^T\hat{B})^{-1}z} \tag{3.70}$$

The relationship of tuning parameters (ζ_L , λ_{max} , z) are evaluated in Equation (3.70). The bandwidth for the *S* dynamics is given by Equation (3.64), in which the spring constant ω_n^2 is the only control parameter for the bandwidth. Equation (3.70) provides two other control parameters (ζ_L , z) which give more capability of tuning and shaping the *S* dynamics.

In summary, the versatile sliding control provides a second-order low-pass filter (S dynamics) and a high-pass (or band-pass) filter (error dynamics). Figure 9 illustrates a detailed synthesis view of versatile sliding control system. The S dynamics as low-pass filter is designed to reject the high-frequency uncertainties (unmodeled dynamics, error dynamics and disturbances). Low-frequency uncertainties are then filtered by high-pass (or band-pass) filter. A detailed block diagram of versatile sliding control system is shown in Figure 10.

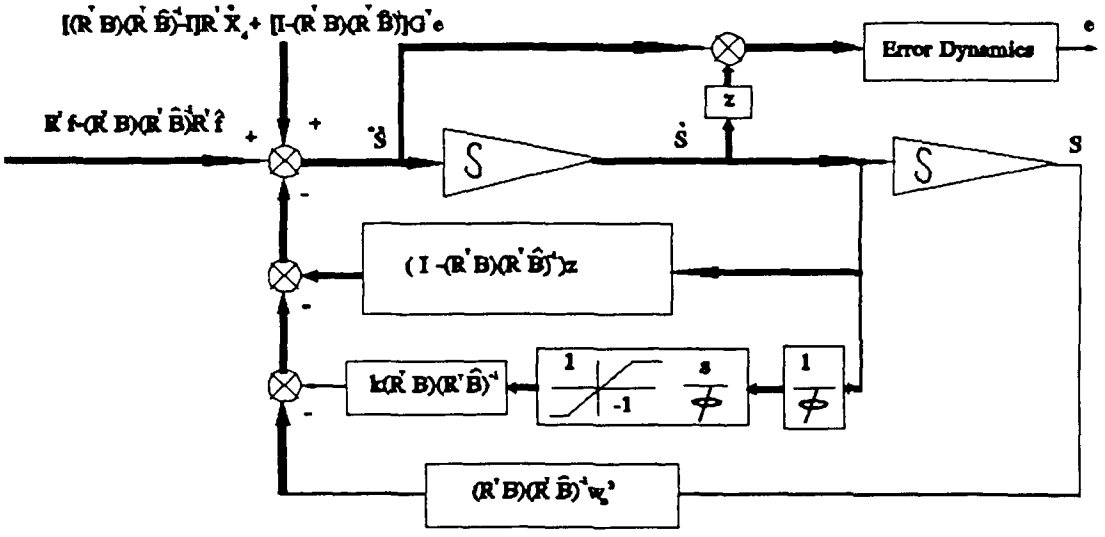

Figure 9 A Detailed Synthesis View of Versatile Sliding Control

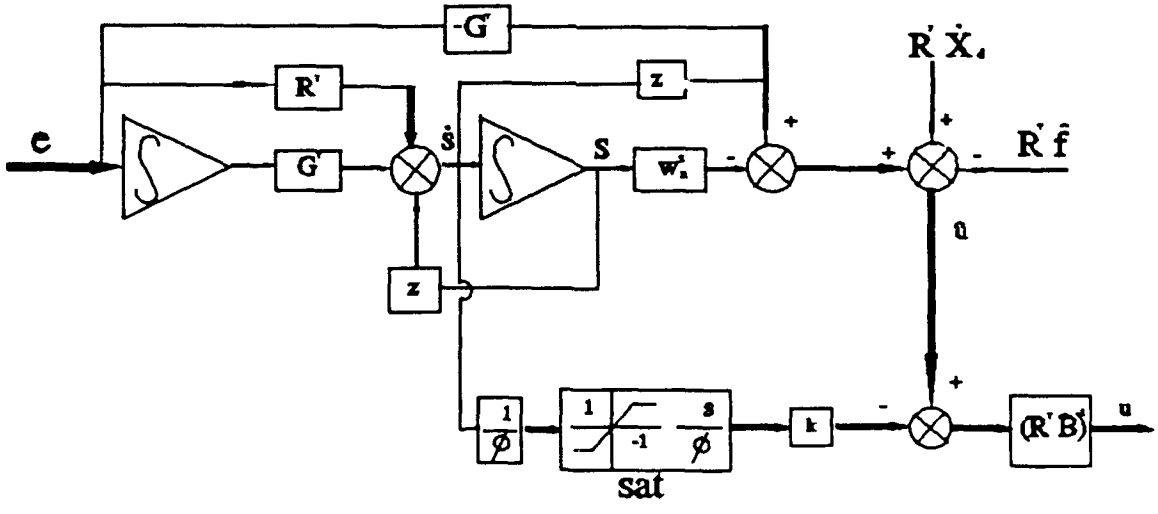

Figure 10 A Detailed Block Diagram of Versatile Sliding Control

IV. RESULTS

To perform simulation and implementation, the structured uncertainties were first to be determined. The nominal values of the system parameters and their bounds were estimated. A non-robust controller was designed to estimate the nominal values and the bounds [Ref. 14]. The time history of these two parameters for two different payload (0 kg and 0.85 kg) illustrated in Figure 11(a)-11(g). Thus, the value for each parameter can be determined and $\hat{B}$, β, $\hat{f}$, and γ can be found in the Table 2.

Table 2

Payload	β	$\hat{B}$	γ	$\hat{f}$
0.00 kg	295.00	$[0\ 0\ 11005.00]^T$	677177.18	$[0.9022\ 10.7036\ -57732.00]^T$
0.85 kg	218.95	$[0\ 0\ 5932.20]^T$	35684.40	$[0.8557\ 5.6229\ -35097.00]^T$

For conservative reasons, set of parameters for the 0.00 kg case was selected and used in the simulation and implementation throughout the whole research. The computer simulation of the control system was performed on a 16 Mhz 80386 IBM compatible personal computer with MATLAB programs. The MATLAB simulation programs and the FORTRAN implementation programs are listed in the appendix A and B.

The SMC control algorithms used for simulation and implementation on the single-link flexible arm are the sliding control with first-order sliding condition, The straight sliding control,

and the versatile sliding control. The following sections will evaluate the performances of these three sliding controllers. To show the robustness to the uncertainties due to varying payload and to perform the tuning procedure such that the unmodeled high-frequency will be rejected are the main goal of this research. The control system performance will be evaluated according to the tracking accuracy, the response speed, and the overshoot.

A. RESULTS ON SLIDING CONTROL WITH FIRST-ORDER SLIDING CONDITION PLUS INTEGRAL ERROR

The system was simulated with a time interval of 0.001 (sec). The controlling parameter η for sliding speed was selected to be 10. An unit-step response of continuous sliding control system of simulation and experiment was first investigated. Figures 12(a)-12(d) show the system responses (i.e., controlled tip position (φ), control input, S response, and boundary layer thickness (ϕ)) of simulation without payload where λ_{max} = 10 (rad/sec). The poles of error dynamics were at p=[-10 -10 -10]. The tip position response shows large overshoot. Time-varying boundary layer thickness (ϕ) was computed to achieve the continuous sliding control. It was observed that $|s_1| < \phi$ and the S dynamics was governed by a first-order dynamics where the controlling parameter λ_{max} determines the response speed. The large overshoot will be minimized by fine tuning the S dynamics such that the system response speed was primarily determined by placing the poles of error dynamics. The tightness of the boundary layer can also improve system response speed.

A higher bandwidth was selected, i.e., λ_{max} = 500 (rad/sec) to reduce the boundary

layer thickness and to reduce the overshoot. Figures 13(a)-13(d) present the performance with no steady-state error and the overshoot was improved. It should be noted that the greater bandwidth will allow high-frequency unmodeled dynamics to excite S dynamics. It was observed that with a greater λ_{max}, the response speed is faster. However, the level of control input is increased. With the tighter boundary layer, the level of overshoot was improved. Because boundary layer thickness becomes thinner and the sampling rate was not high enough, the occurrence of chattering is not surprising and the current (i.e., control input) saturation occurs. By selecting the higher sampling rate to eliminate chattering and saturation, i.e., $\Delta t = 0.0005$ (sec), Figures 14(a)-14(d) show the further improvement of control performance in the simulation without payload. Figures 14(e)-14(h) show the experimental results. Figures 15(a)-15(d) show the responses in simulation and Figures 15(e)-15(h) show those in experiment with payload (0.85 kg) when $\lambda_{max} = 500$ (rad/sec) and simulation time interval $\Delta t = 0.0005$ (sec).

The next test was to examine the tracking performance. The desired trajectory was defined as

$$\varphi = \begin{cases} t & \text{for } 0 \le t < 1 \\ 1 & \text{for } 1 \le t < 2 \\ 3-t & \text{for } 2 \le t < 2.4 \\ 0.6 & \text{for } t > 2.4 \end{cases} \tag{4.1}$$

Figures 16(a)-16(h) show the tracking performance of the system in simulation and experiment without payload and Figures 17(a)-17(h) show the tracking performance with payload (0.85 kg). A greater bandwidth ($\lambda_{max} = 1700$ rad/sec) was selected to tight the boundary layer.

With the trajectory control, the overshoot can be further minimized. The improvement

on reducing the overshoots is very important for robot applications. The overshoot in the experimental results reflected the simulation prediction. The time history of the boundary layer thickness in the simulation differs from that in the experiment. This difference can be neglected since the change of the boundary layer thickness compared to its magnitude was quite small. The controller is robust to varying payload because the system responses were not affected by changing the payload.

In summary, the maximum bandwidth is the only parameter to reject the unmodeled high-frequency dynamics and at the same time to control tracking accuracy in this control algorithm. Therefore, a trade-off exists between the robustness to the unmodeled dynamics and the tracking accuracy. With an integral control in error dynamics, the steady-state error was eliminated. The controller is indeed robust to the uncertainties due to varying payload.

B. RESULTS ON STRAIGHT SLIDING CONTROL

Step response will first be studied to examine the performance of the straight sliding control. An equivalent spring constant and a damping ratio were selected as $\omega_n = 10$ (rad/sec) and $\zeta_L = 1$ simulated at $\Delta t = 0.0002$ (sec), while the poles of error dynamics were selected as p=[-10 -10]. Figures 18(a)-18(d) show results of a step response, the control input, and the S and ϕ of the simulation without payload. The control presents not only overshoot but also long settling time. Thus, damping ratio (ζ_L) will be used to fine tune S dynamics such that the overshoot will be minimized and the settling time will be improved. The equivalent spring constant (ω_n^2) will be lowered to reject high-frequency excitation of the S dynamics.

The effects due to the damping ratio ζ_L will be examined. The damping ratio $\zeta_L = 10$ was selected to reduce the boundary layer thickness while $\omega_n = 10$ (rad/sec). Figures 19(a)-19(d) show the performance of step response, the control input, and the S and ϕ of the simulation without payload. Figures 19(e)-19(h) show the step response of experiment at $\zeta_L = 10$ and $\omega_n = 10$ (rad/sec). The overshoot was minimized and the response speed was faster than before.

The next test is to lower the ω_n^2, i.e., lowering the bandwidth of the S dynamics to reject the high-frequency excitations. $\omega_n = 1$ (rad/sec) and $\zeta_L = 10$ were selected. Figures 20(a)-20(d) present the control performance (without payload) where the step response has steady-state error. To reject the unwanted high-frequency excitations, the control system has to pay the price. However, the damping ratio ζ_L can be increased to remedy the sluggish of the response. Figures 21(a)-21(h) show the control performance of $\omega_n = 1$ (rad/sec) and $\zeta_L = 100$ for both simulation and experiment without payload. Figures 22(a)-22(h) present the control performance of $\omega_n = 1$ (rad/sec) and $\zeta_L = 100$ for both simulation and experiment with payload (0.85 kg). It is noted that the control performance showed no difference in changing the payload.

The last investigation was to examine robustness to varying payload of the tracking performance. The desired trajectory was designed as in Equation (4.1). Figures 23(a)-23(d) show the tracking performance for the simulation at $\omega_n = 1$ (rad/sec) and $\zeta_L = 100$ without payload.

Figures 23(e)-23(h) show the tracking performance for the experiment at ω_n = 1 (rad/sec) and ζ_L = 100 without payload. Figures 24(a)-24(h) illustrate the tracking performance for the simulation and experiment at ω_n = 1 (rad/sec) and ζ_L = 100 with payload (0.85 kg). The ω_n = 1 (rad/sec) was selected not to excite the unmodeled high-frequency dynamics.

Note that the control performance was not affected by the payload. The greater the damping ratio ζ_L, the better the tracking performance and the greater the control effort is needed. Although the smoothed control law was used to compute the current required to the actuator, a little chattering occurred in the experimental results while the arm was trying to stand still.

In summary, the straight sliding control utilizes the first-order plus integral sliding condition to provide tracking accuracy, disturbance rejection, and stability. The S dynamics plays a key role to obtain a desired system response since the error dynamics is affected by the output of S dynamics. The two control parameters ζ_L and ω_n^2 are adjusted to accomplish the control task without conflicting. The damping ratio ζ_L was selected to tight the S dynamics performances, while the ω_n^2 in the sliding condition to keep the excitation from entering into the S dynamics. Therefore, the trade-off between tracking accuracy and robustness to uncertainties (due to unmodeled high-frequency dynamics) has no longer exist. Also, the robustness to the varying payload was proved by the simulation and the experiment.

C. RESULTS ON VERSATILE SLIDING CONTROL

The step response of versatile sliding control was to test the control performances. To keep the bandwidth of S dynamics low, the ω_n^2 was selected to be 1. The zero (z) not only provides the filtering level of error dynamics but also affects the boundary layer thickness. Thus, the z should be chosen carefully such that the S dynamics can be fine tuned. For a given value of ζ_L, the smaller z, the larger boundary layer thickness. Also, greater boundary layer thickness gives more active S dynamics. Given $z = 1$ and $\zeta_L = 10$, Figures 25(a)-25(d) show the step responses, the control input, and the $\dot{S}$ and ϕ of simulation, while the poles of error dynamics were p=[-10 -10 -10]. Figures 25(e)-25(h) show the experiment results. The overshoot exhibited in the control motion. With the same ζ_L, Figures 26(a)-26(d) present the responses of a greater value $z = 10$ without payload. The overshoot was reduced by reducing the boundary layer thickness. However, the steady-state error occurred. Figures 26(e)-26(h) present the results for the experiment without payload at $z = 10$ and $\zeta_L = 10$. Figures 27(a)-27(h) present the simulation and the experiment with payload (0.85 kg) at $z = 10$ and $\zeta_L = 10$.

The next test was to examine tracking performance. The trajectory was designed as in Equation (4.1). Given $z = 10$, $\zeta_L = 10$, and $\omega_n^2 = 1$ (rad/sec), Figures 28(a)-28(h) show the performances of simulation and experiment.

The overall tendency of the control system in the experiment has the same results as expected in the simulation. The smaller the boundary layer, the smaller the steady-state error,

and the greater control input is needed. The little chattering occurred in the straight sliding control was seen in the performance of versatile sliding control. It is noted that the robust to the varying payload is achieved by increasing the payload while the motion control will not be changed.

In summary, the versatile sliding control provides three parameters to accomplish the control motion. The damping ratio ζ_L and z were chosen to tight the boundary layer thickness and also the ω_n^2 was set low to reject the excitations entering the error dynamics. The z not only gives the tuning on the boundary layer thickness but also make the error dynamics a band-pass (or high-pass) filter. Actually, the versatile sliding control provides more tuning parameters which make the trade-off between tracking accuracy and robustness to uncertainties (unmodeled high-frequency dynamics) no longer exist. Also, the versatile sliding controller is proved to be robust to the uncertainties due to varying payload.

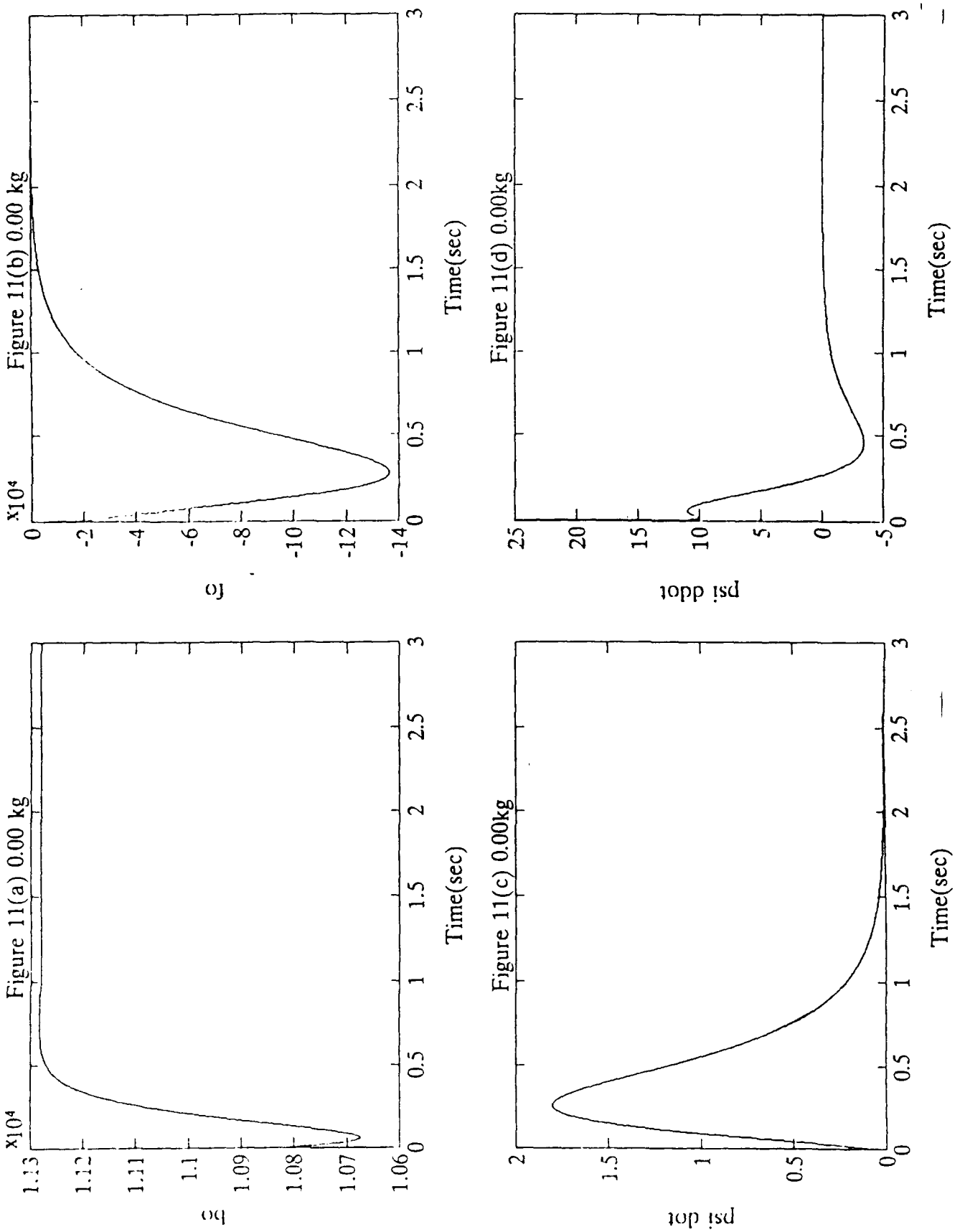

Figure 11(a)-(d) The Time History of System Parameters

Figure 11(e)-(h) The Time History of System Parameters

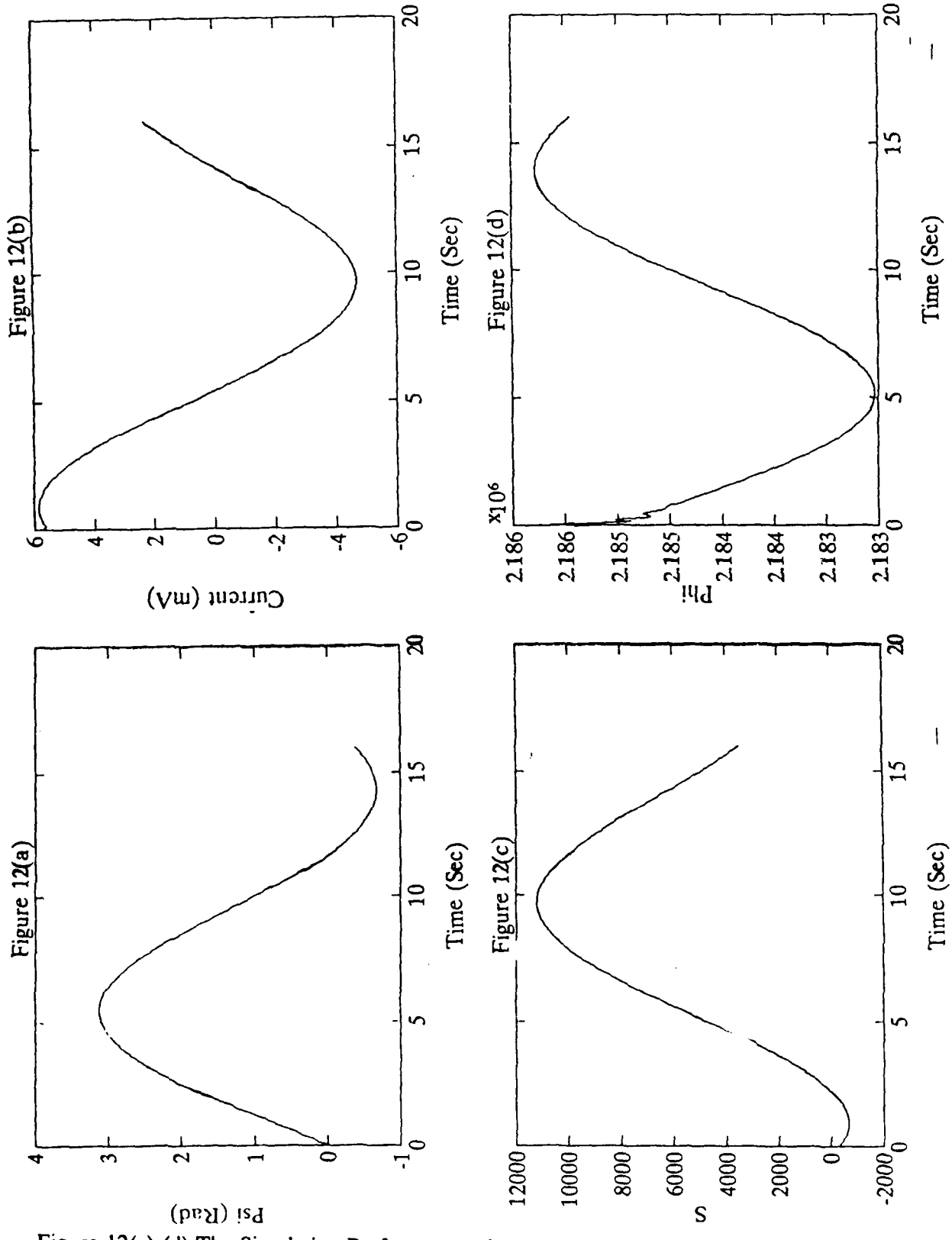

Figure 12(a)-(d) The Simulation Performance of SMC with First-Order Sliding Condition (λ_{max} = 10 rad/sec, Δt = 0.001 (sec), payload = 0.00 kg)

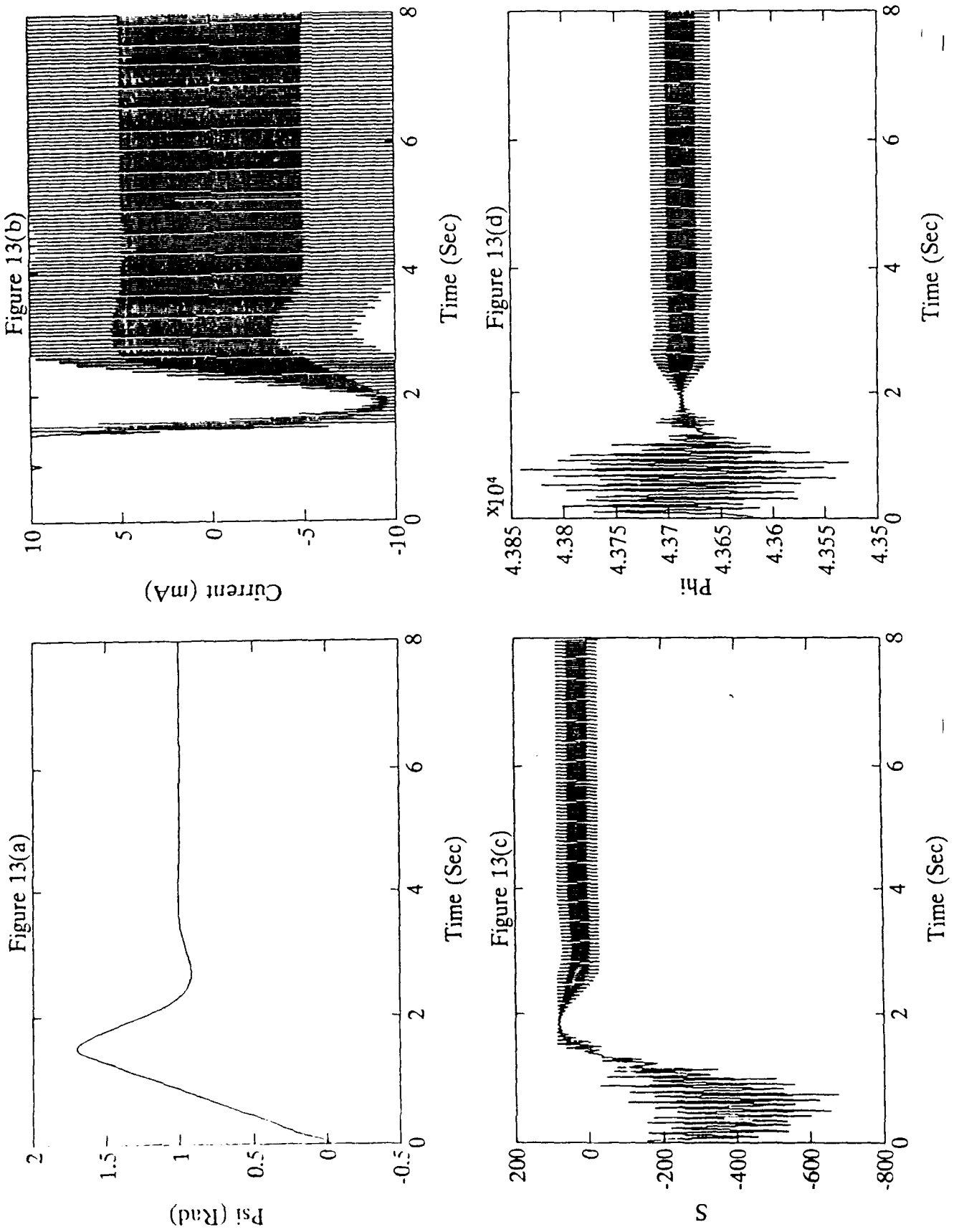

Figure 13(a)-(d) The Simulation Performance of SMC with First-Order Sliding Condition (λ_{max} = 500 rad/sec, Δt = 0.001 (sec), payload = 0.00 kg)

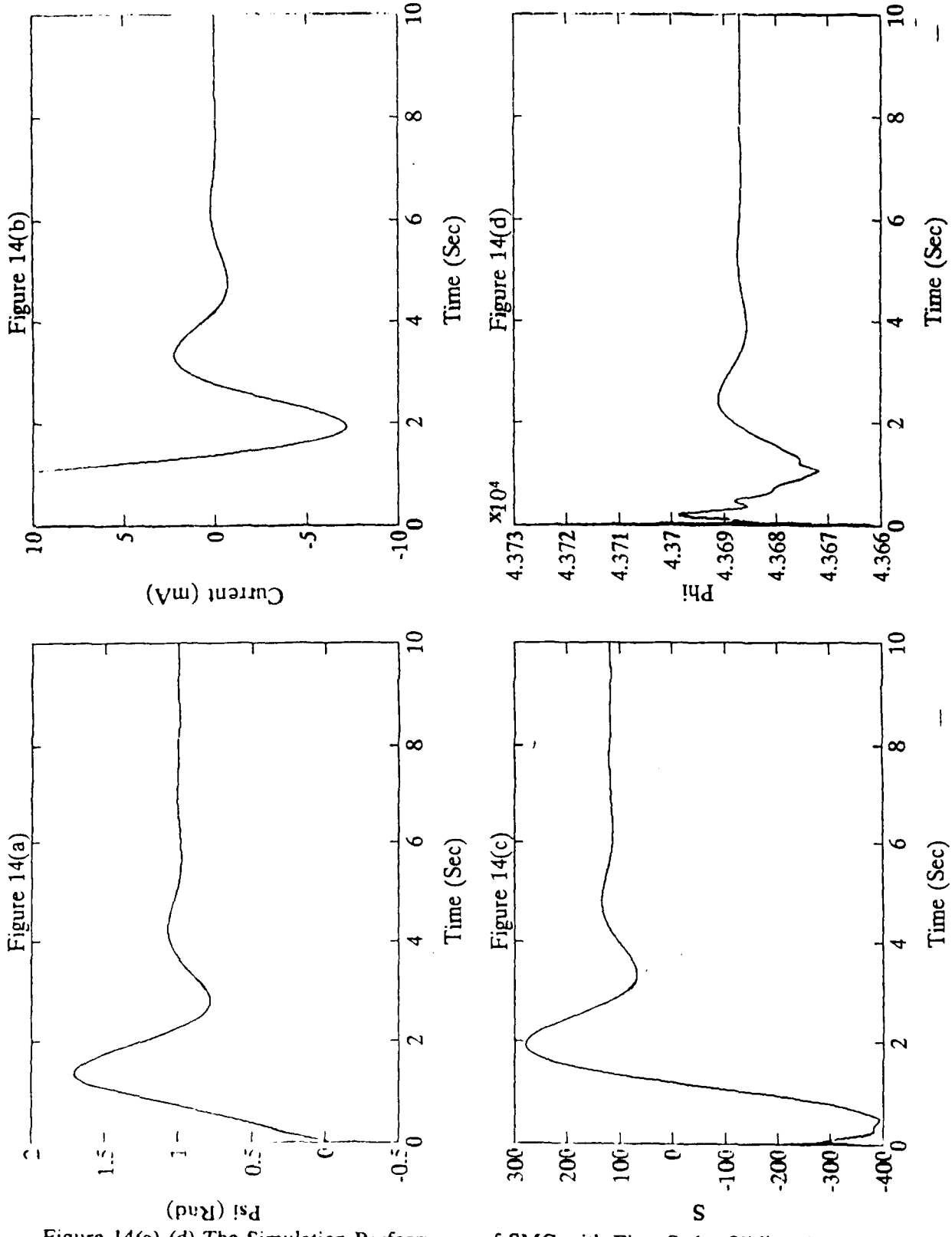

Figure 14(a)-(d) The Simulation Performance of SMC with First-Order Sliding Condition (λ_{max} = 500 rad/sec, Δt = 0.0005, (sec), payload = 0.00 kg)

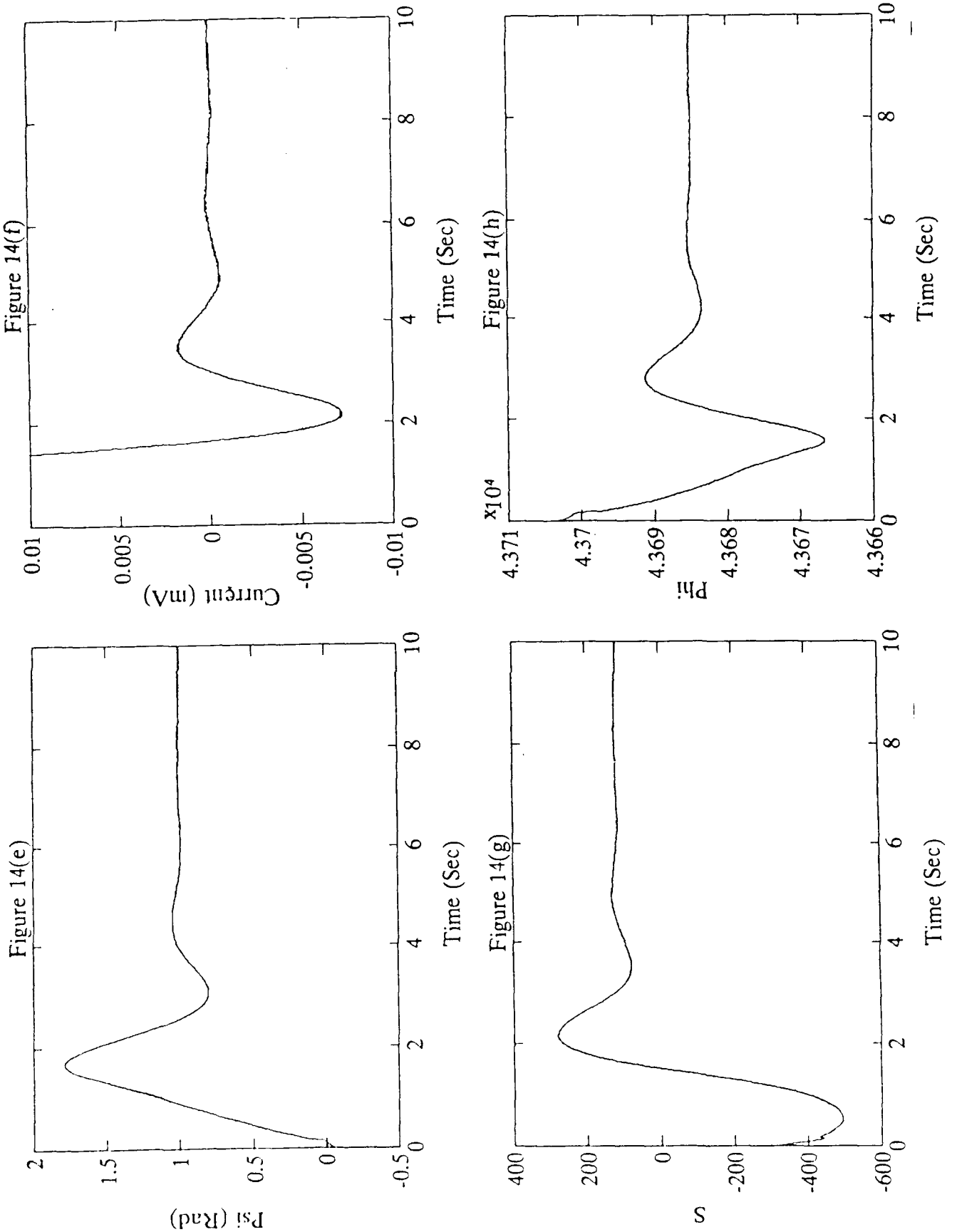

Figure 14(e)-(h) The Experiment Performance of SMC with First-Order Sliding Condition (λ_{max} = 500 rad/sec, payload = 0.00 kg)

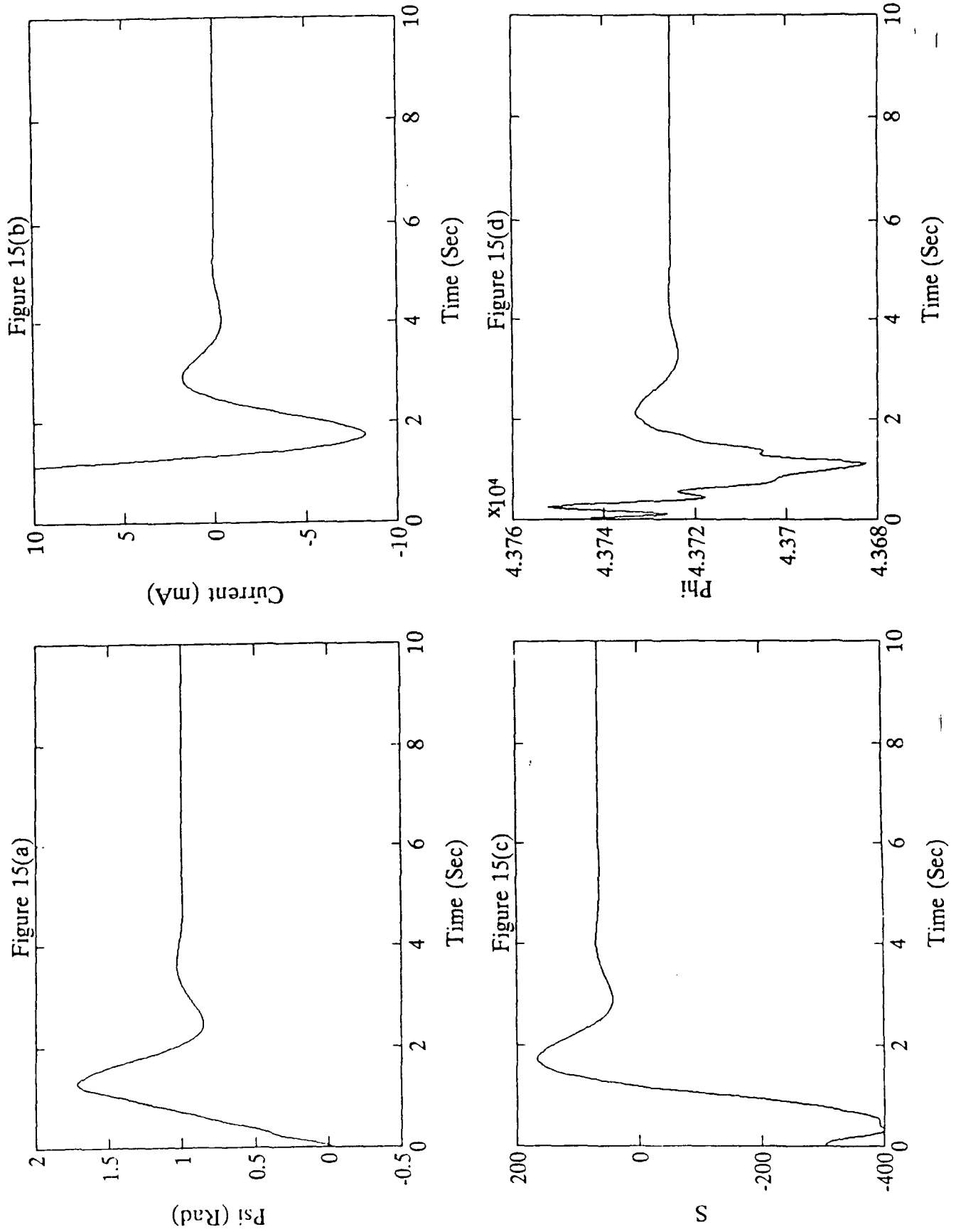

Figure 15(a)-(d) The Simulation Performance of SMC with First-Order Sliding Condition (λ_{max} = 500 rad/sec, Δt = 0.0005 sec, payload = 0.85 kg)

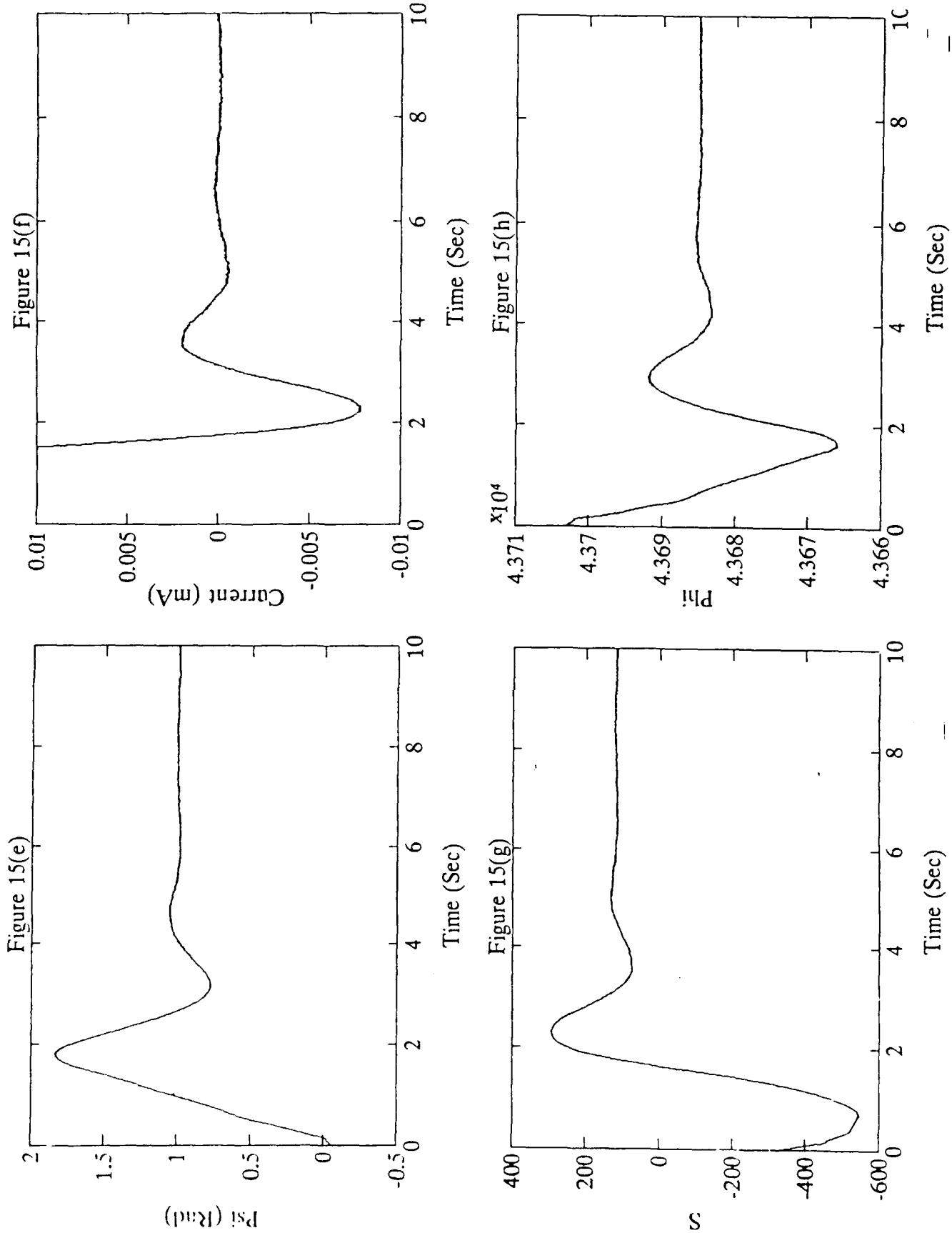

Figure 15(e)-(h) The Experiment Performance of SMC with First-Order Sliding Condition (λ_{max} = 500 rad/sec payload = 0.85 kg)

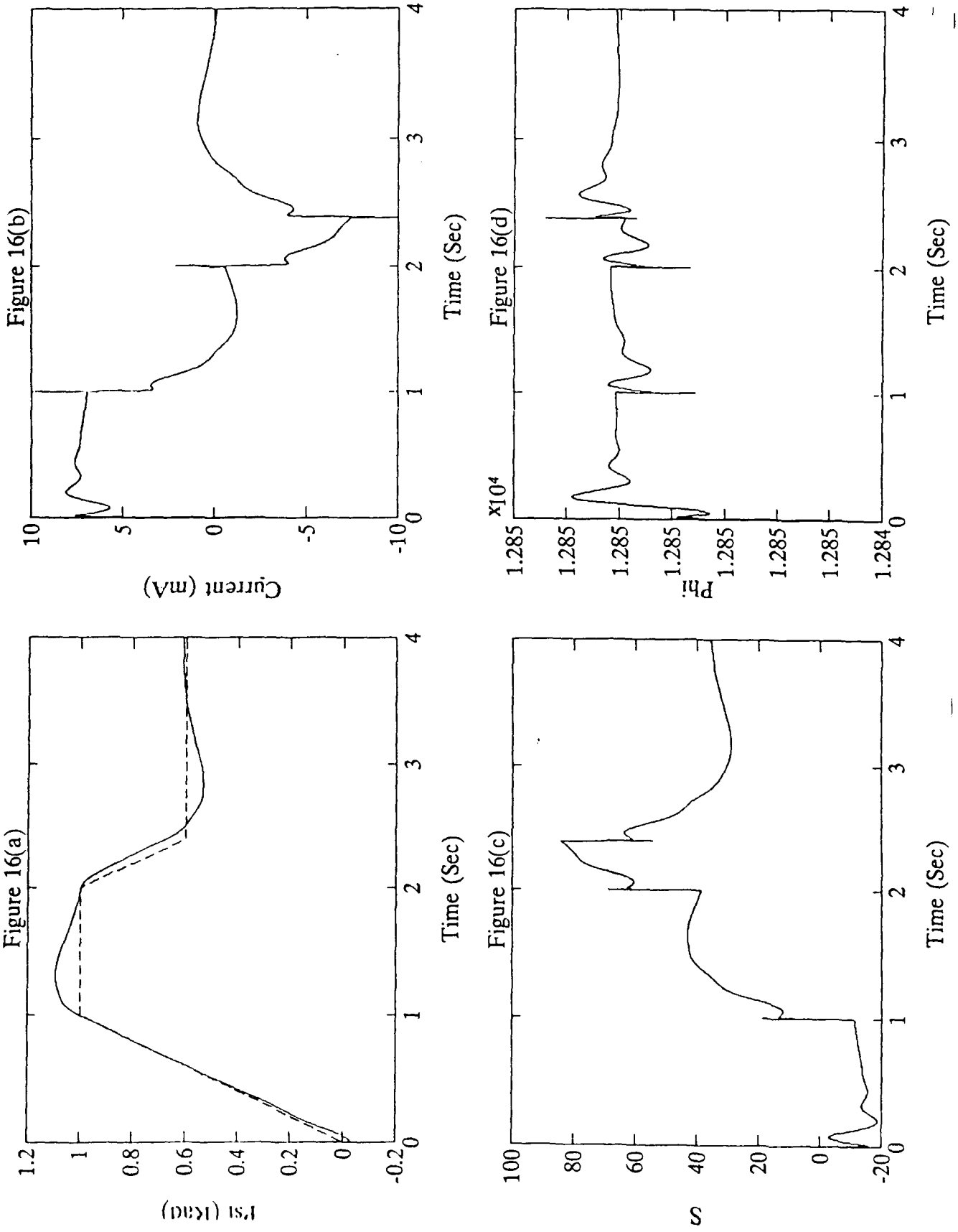

Figure 16(a)-(d) The Simulation of Tracking Performance of SMC with First-Order Sliding Condition (λ_{max} = 1700 rad/sec, Δt = 0.0005 (sec), payload = 0.00 kg)

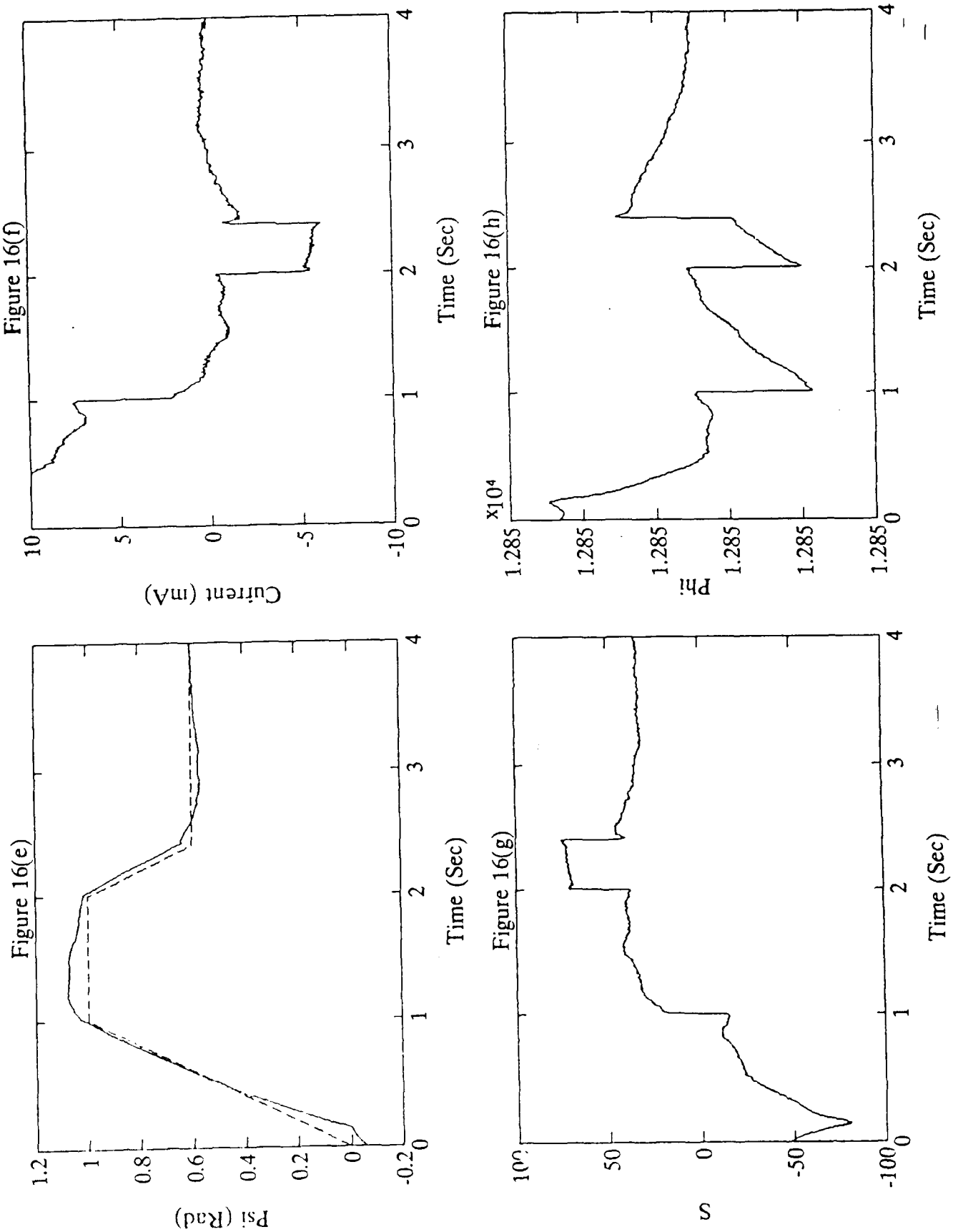

Figure 16(e)-(h) The Experiment of tracking Performance of SMC with First-Order Sliding Condition (λ_{max} = 500 rad/sec, payload = 0.00 kg)

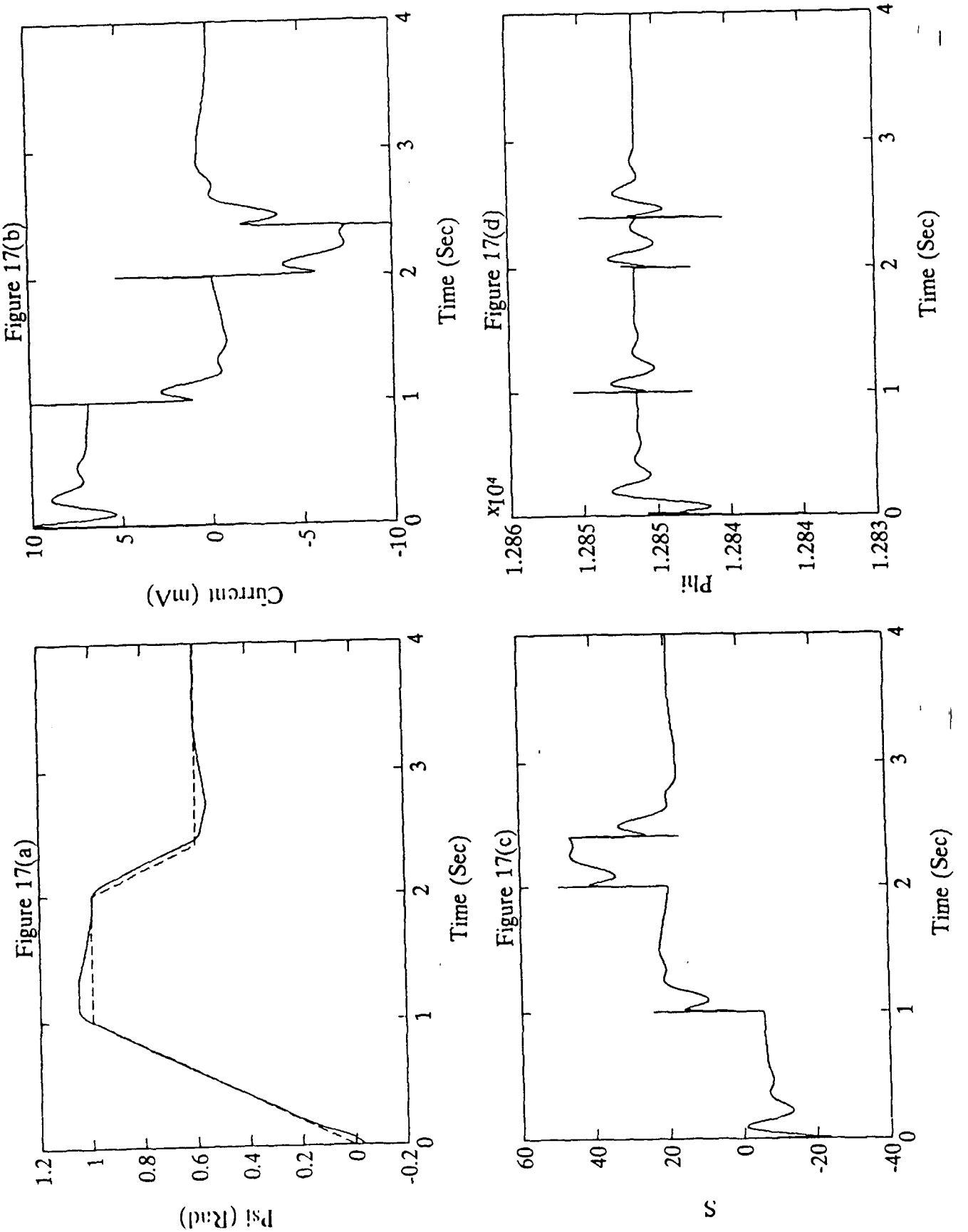

Figure 17(a)-(d) The Simulation of tracking Performance of SMC with First-Order Sliding Condition (λ_{max} = 1700 rad/sec, Δt = 0.0005 (sec), payload = 0.85 kg)

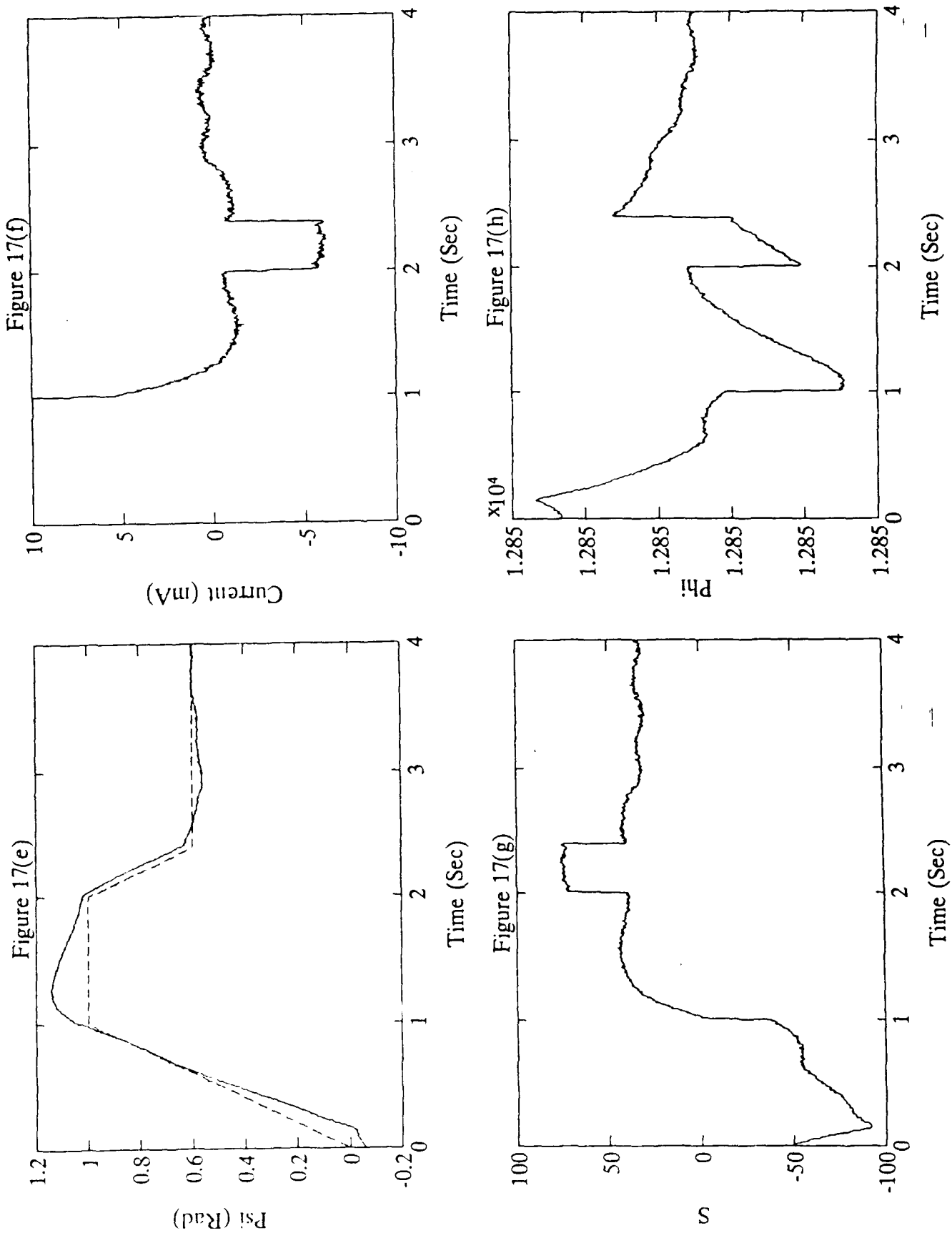

Figure 17(e)-(h) The Experiment of tracking Performance of SMC with First-Order Sliding Condition (λ_{max} = 1700 rad/sec, payload = 0.00 kg)

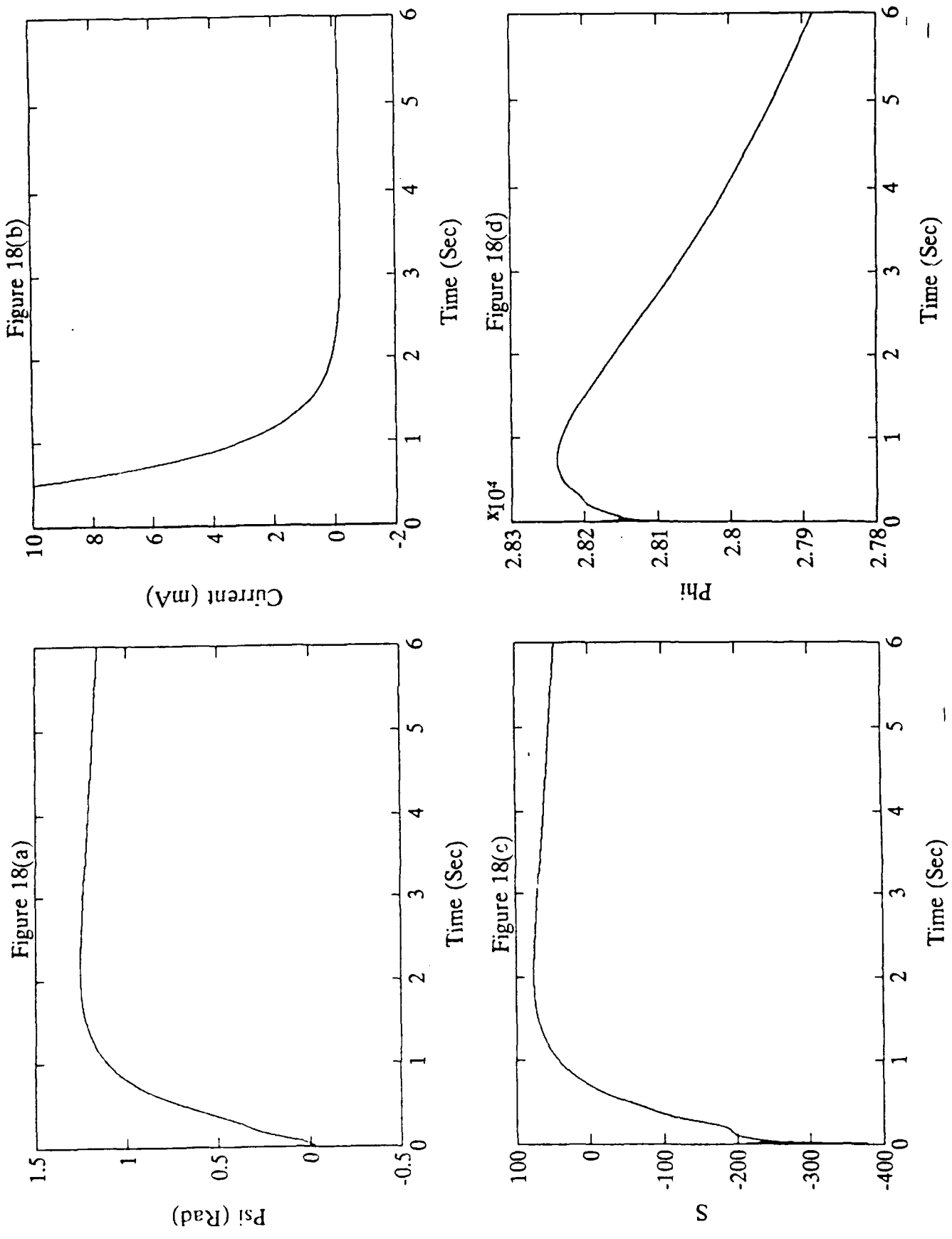

Figure 18(a)-(d) The Simulation Performance of Straight Sliding Control
(ω_n = 10 rad/sec, ζ_L = 1, Δt = 0.0002 (sec), payload = 0.00 kg)

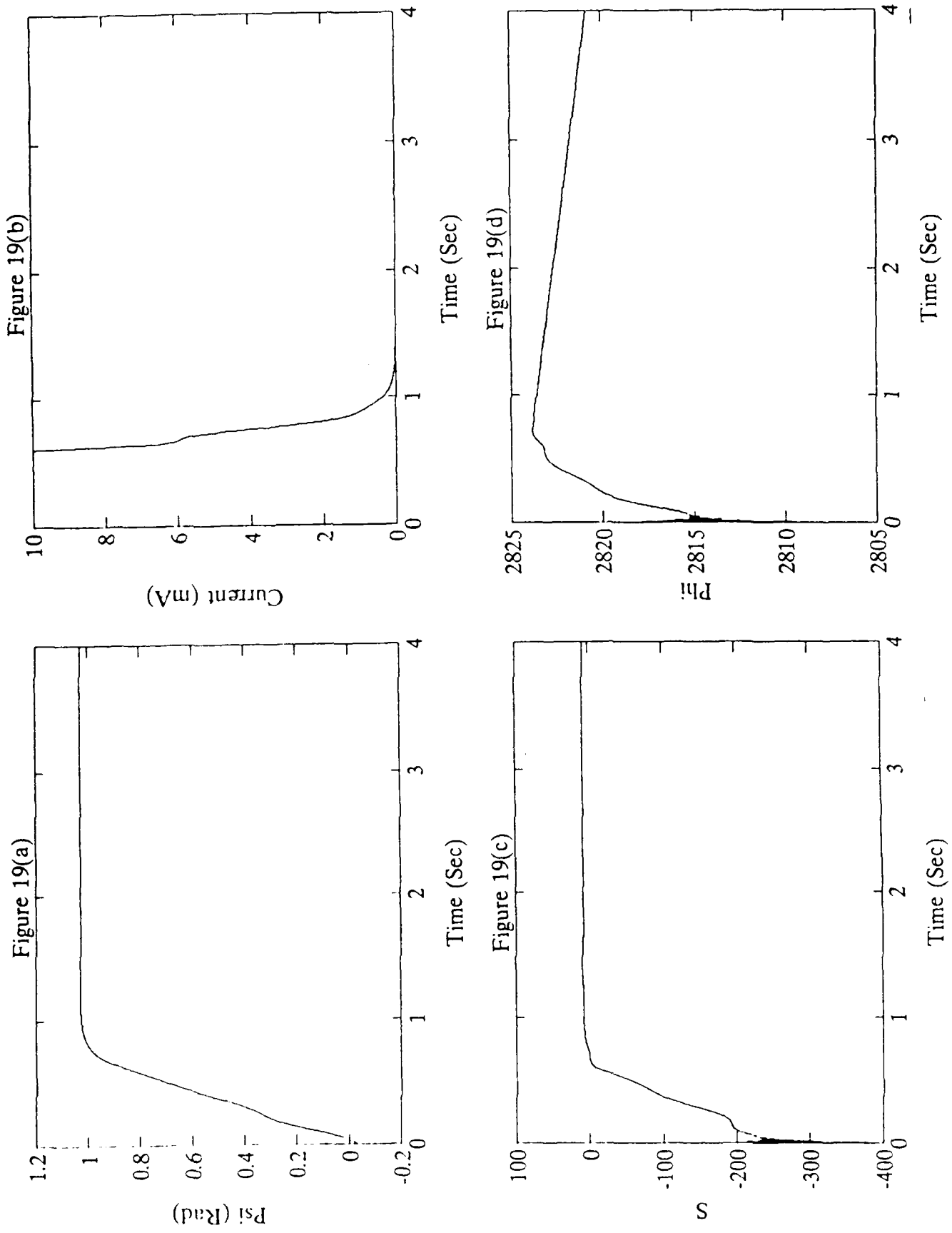

Figure 19(a)-(d) The Simulation Performance of Straight Sliding Control
(ω_n = 10 rad/sec, ζ_L = 10, Δt = 0.0002 (sec), payload = 0.00 kg)

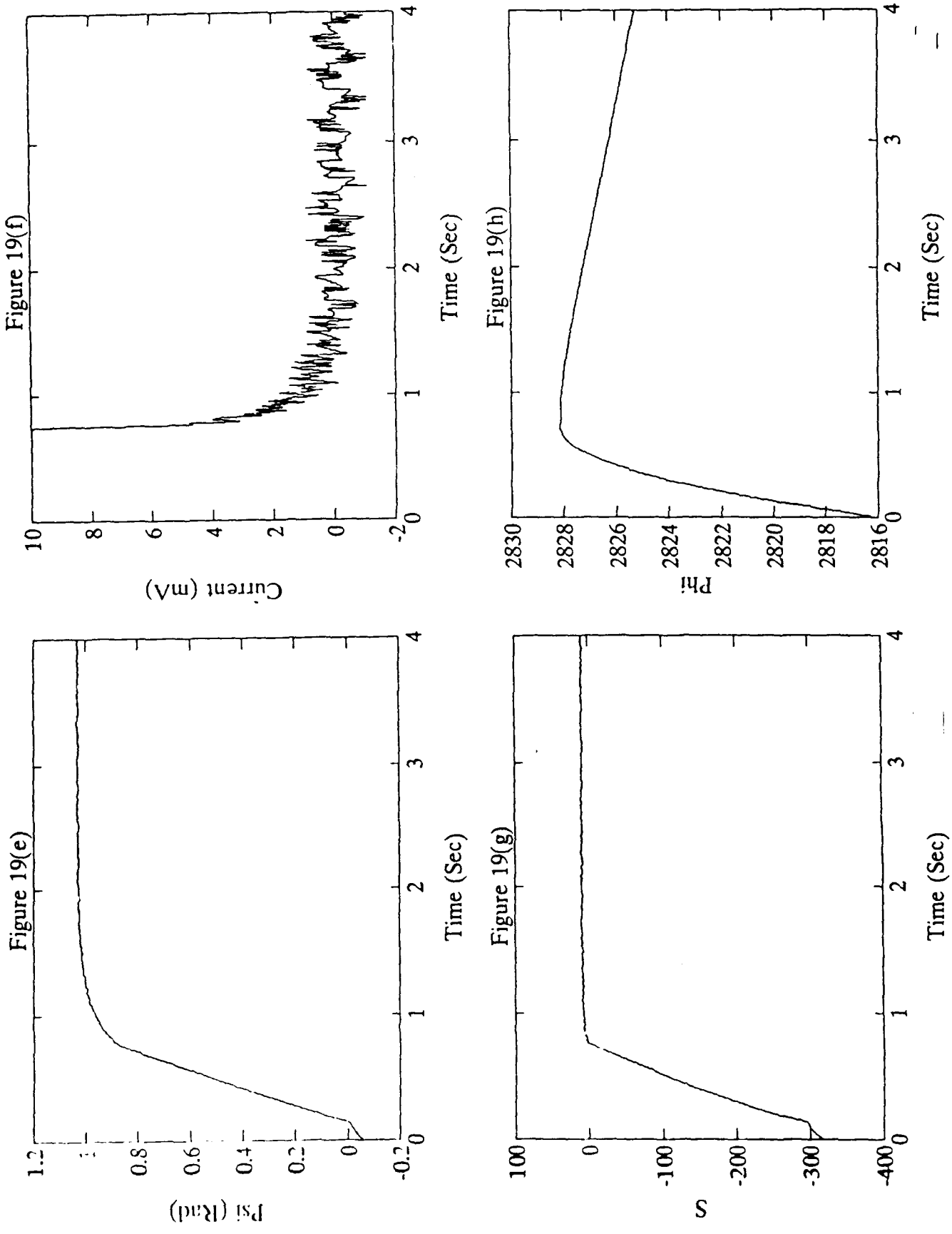

Figure 19(e)-(h) The Experiment Performance of Straight Sliding Control
(ω_n = 10 rad/sec, ζ_L = 10, payload = 0.00 kg)

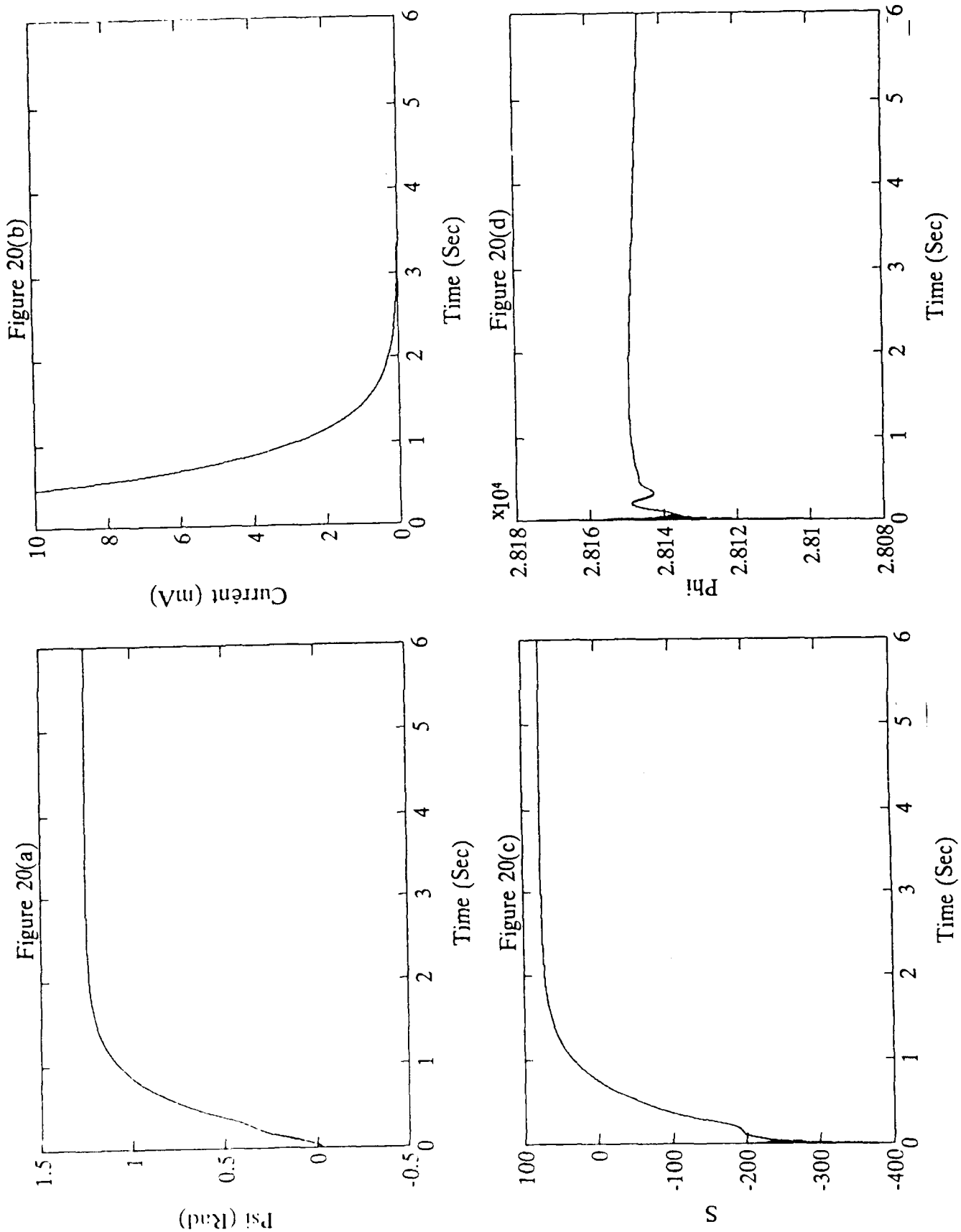

Figure 20(a)-(d) The Simulation Performance of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 10, Δt = 0.0002 (sec), payload = 0.00 kg)

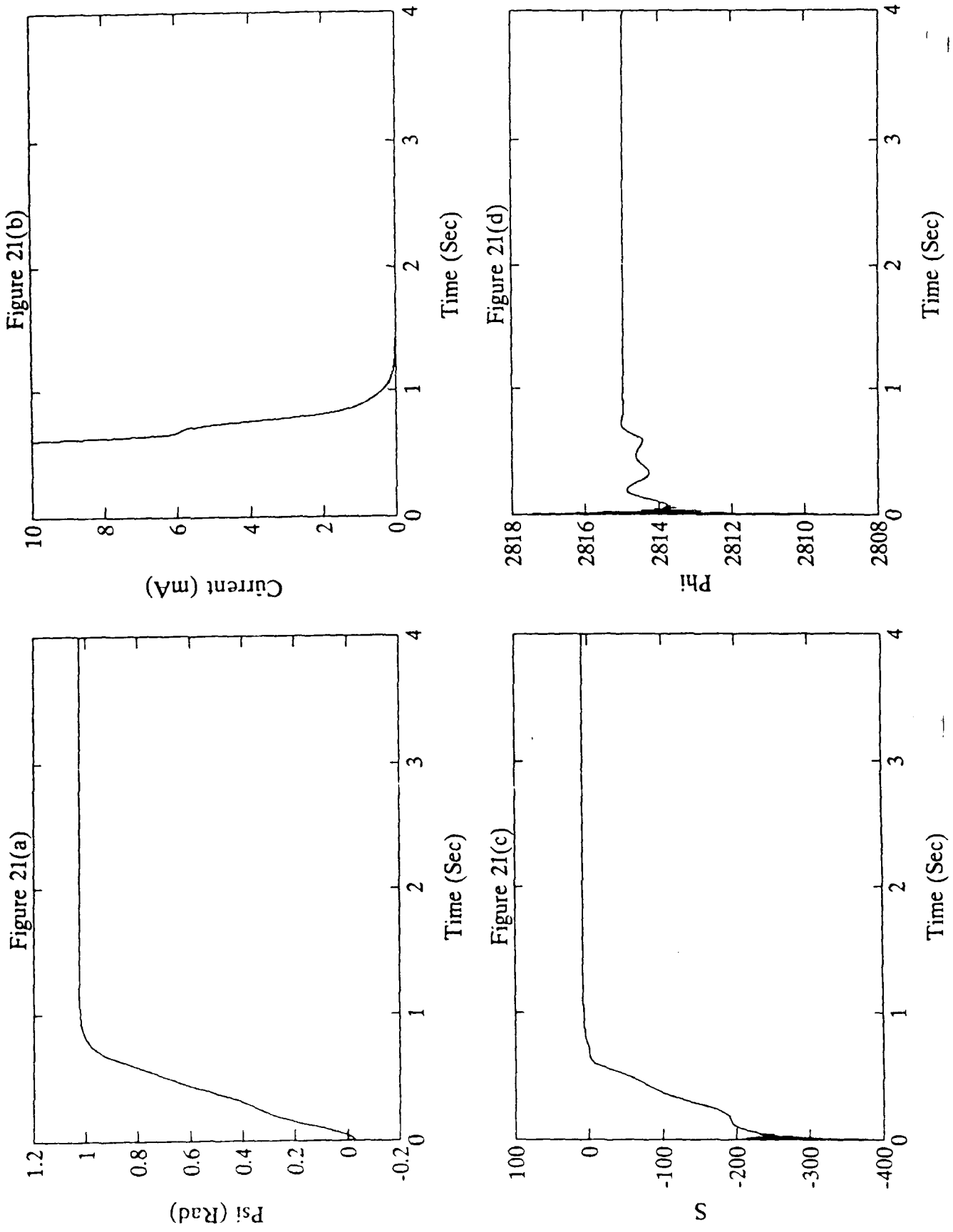

Figure 21(a)-(d) The Simulation Performance of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, Δt = 0.0002 (sec), payload = 0.00 kg)

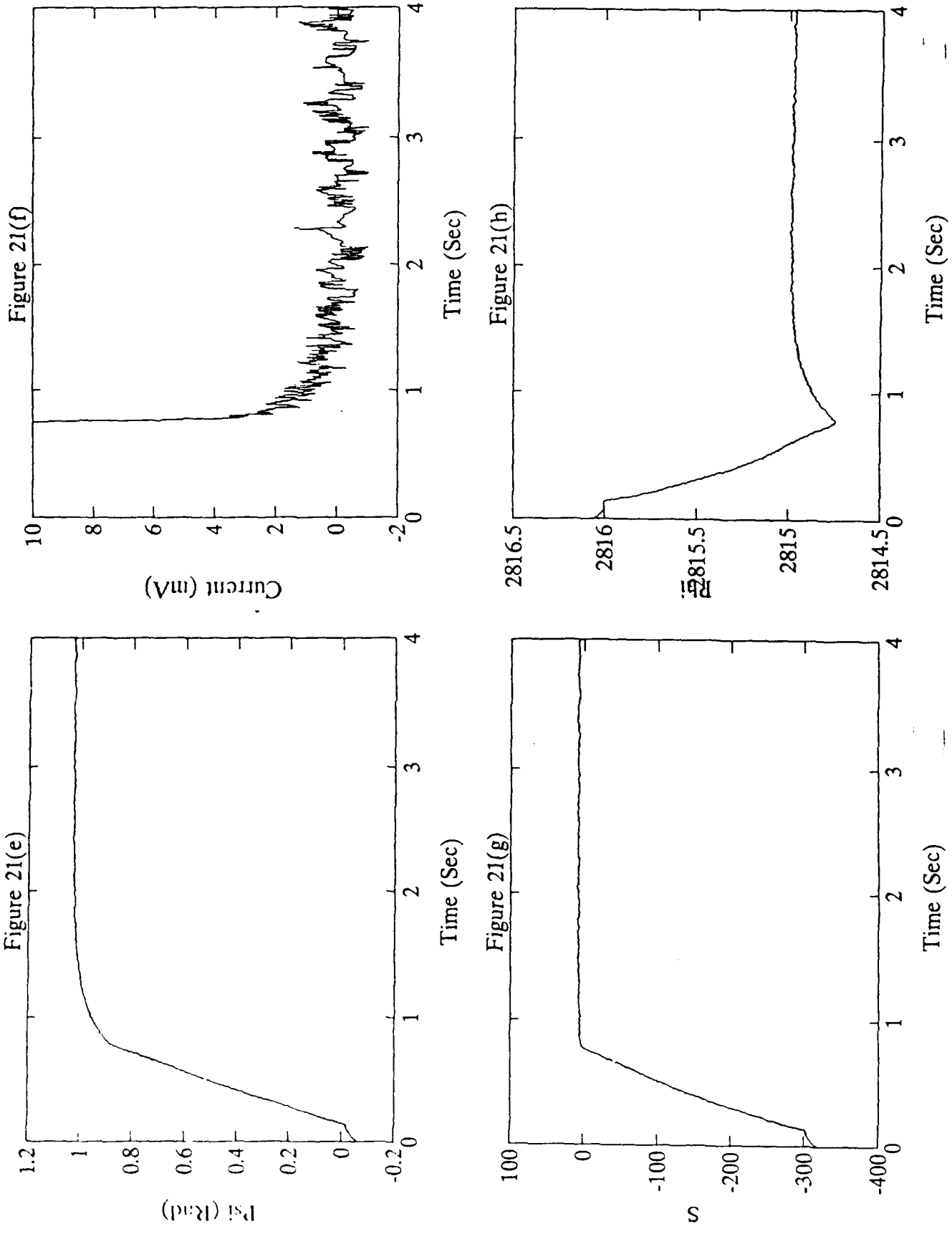

Figure 21(e)-(h) The Experiment Performance of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, payload = 0.00 kg)

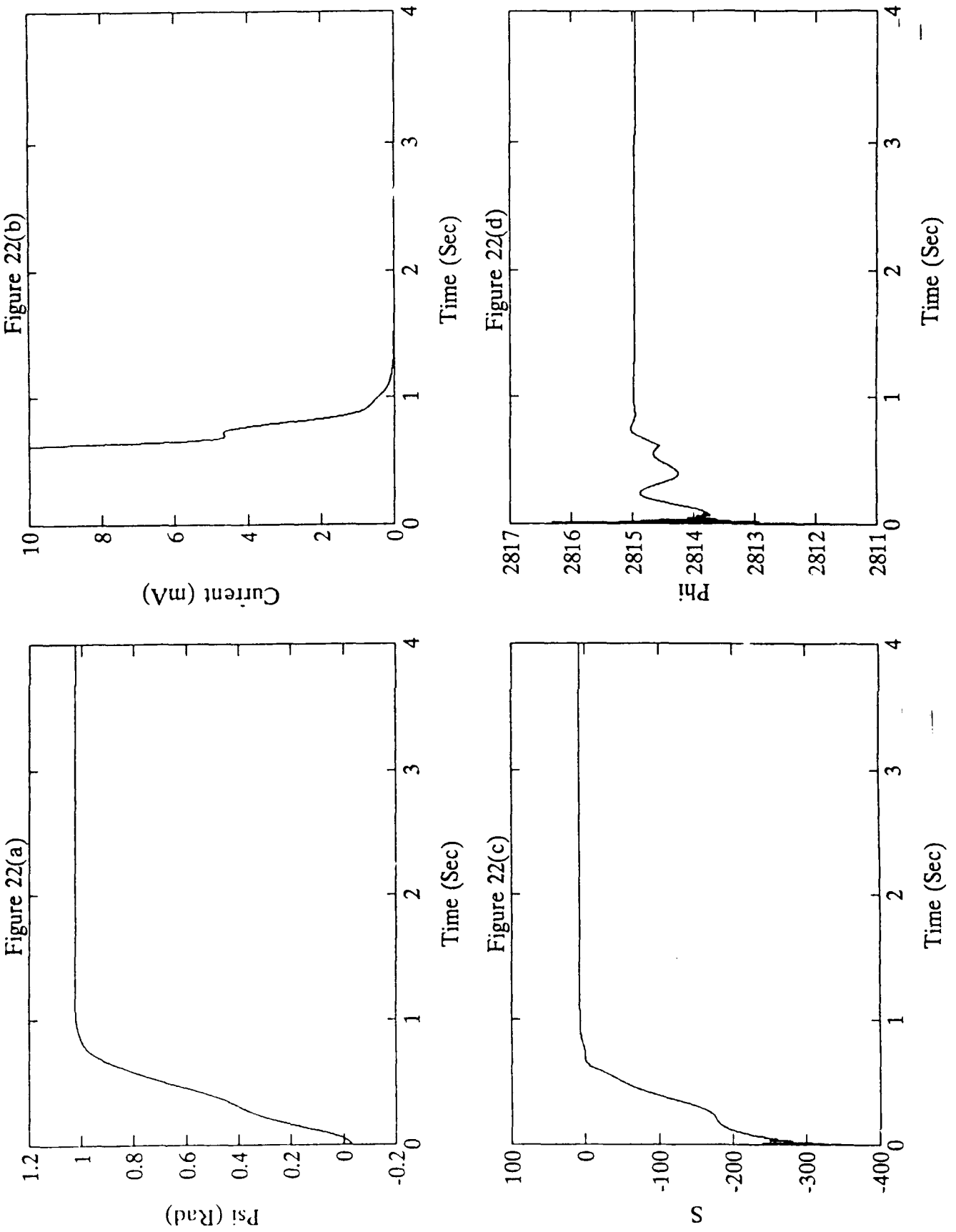

Figure 22(a)-(d) The Simulation Performance of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, Δt = 0.0002 (sec), payload = 0.85 kg)

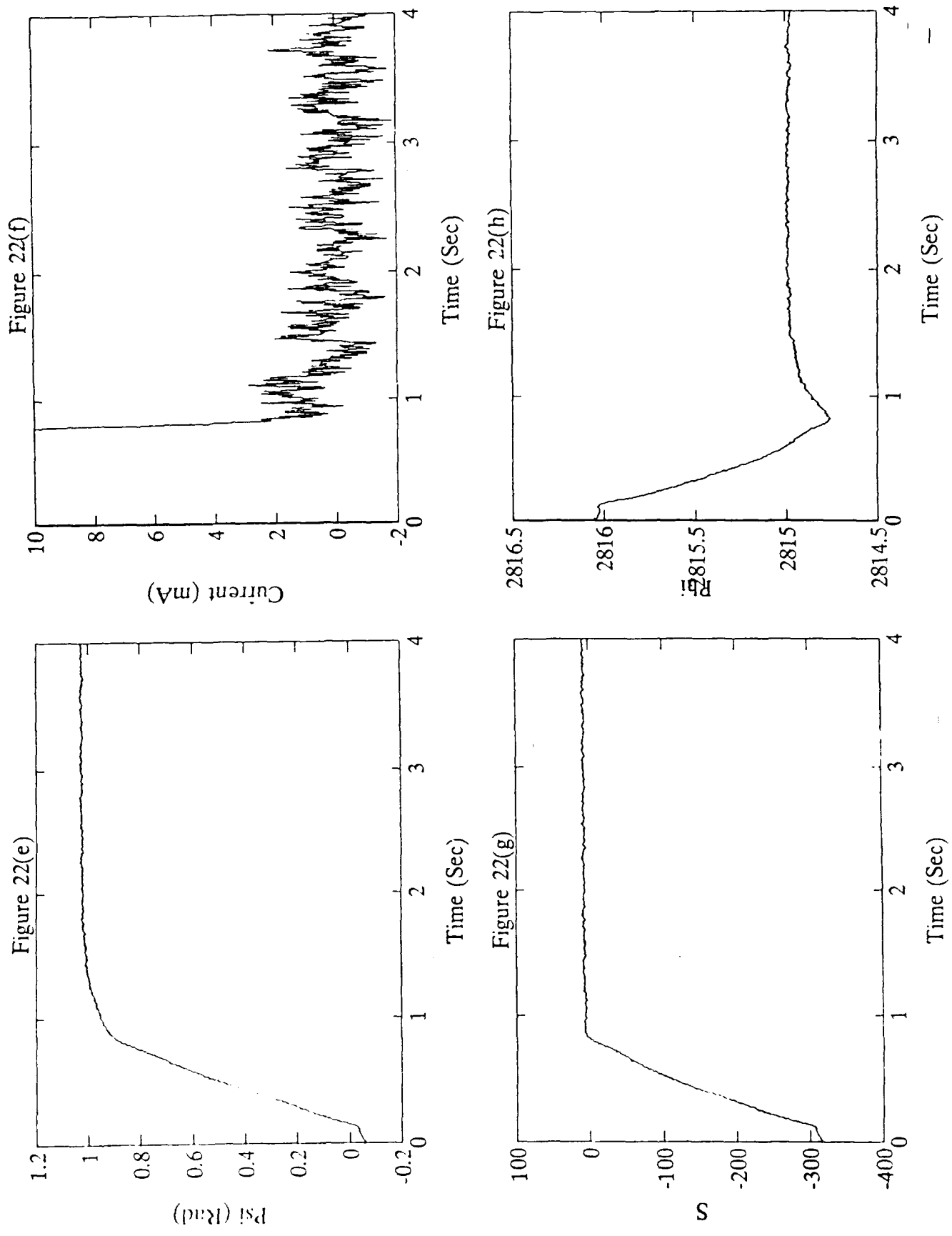

Figure 22(e)-(h) The Experiment Performance of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, payload = 0.85 kg)

Figure 23(a)-(d) The Tracking Performance in Simulation of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, Δt = 0.0002 (sec), payload = 0.00 kg)

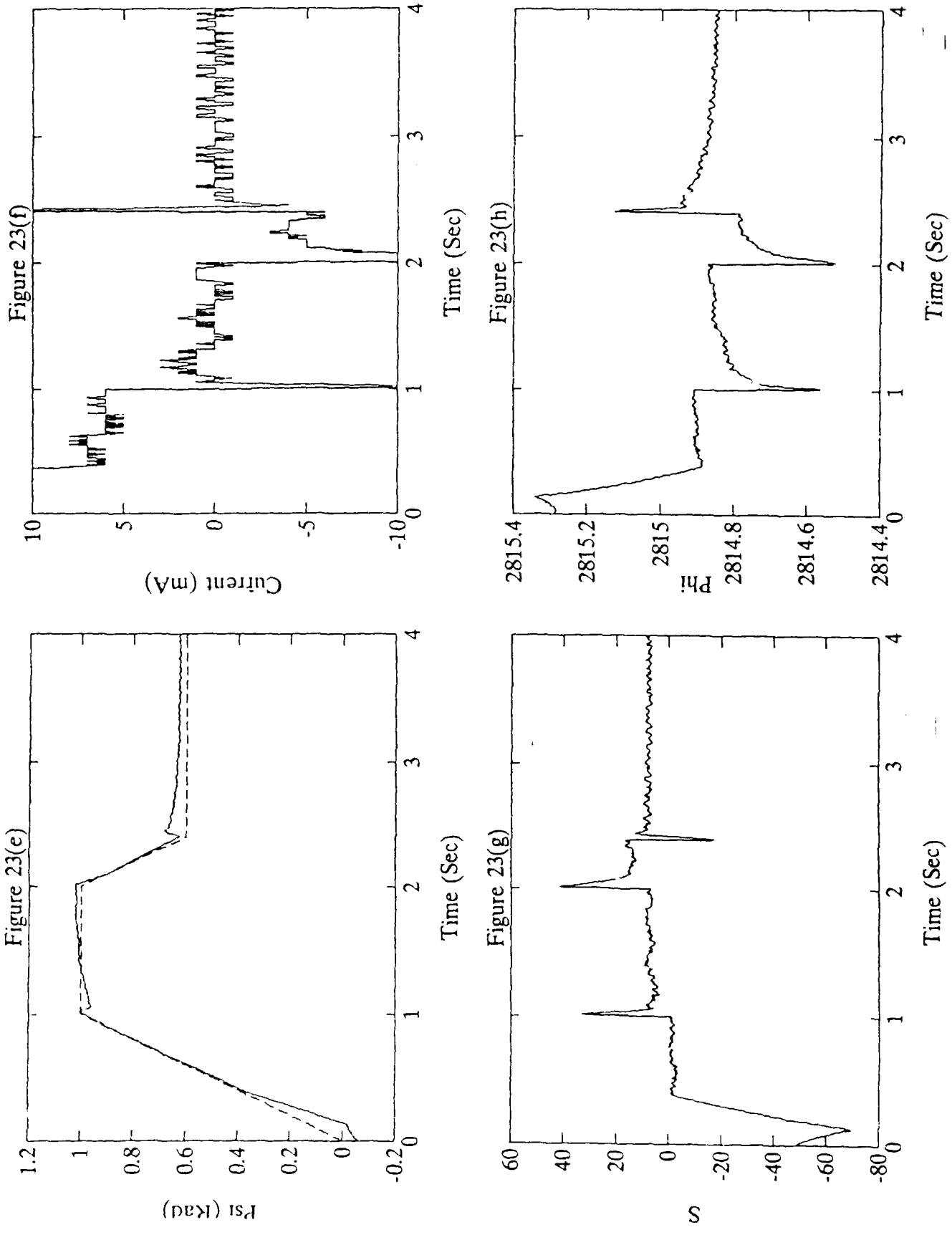

Figure 23(e)-(h) The Tracking Performance in Experiment of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, payload = 0.00 kg)

Figure 24(a)-(d) The Tracking Performance in Simulation of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, Δt = 0.0002 (sec), payload = 0.85 kg)

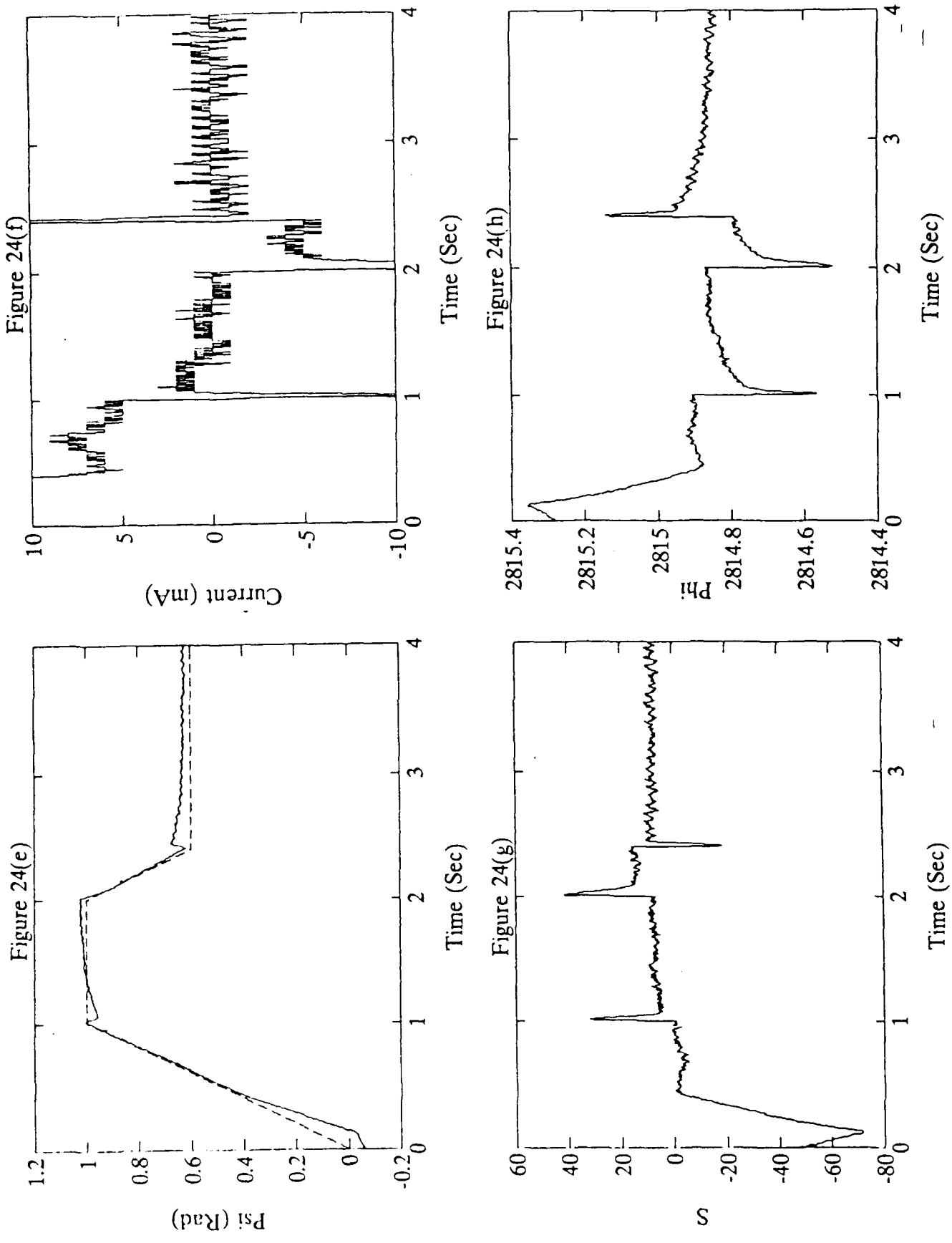

Figure 24(e)-(h) The Tracking Performance in experiment of Straight Sliding Control
(ω_n = 1 rad/sec, ζ_L = 100, payload = 0.85 kg)

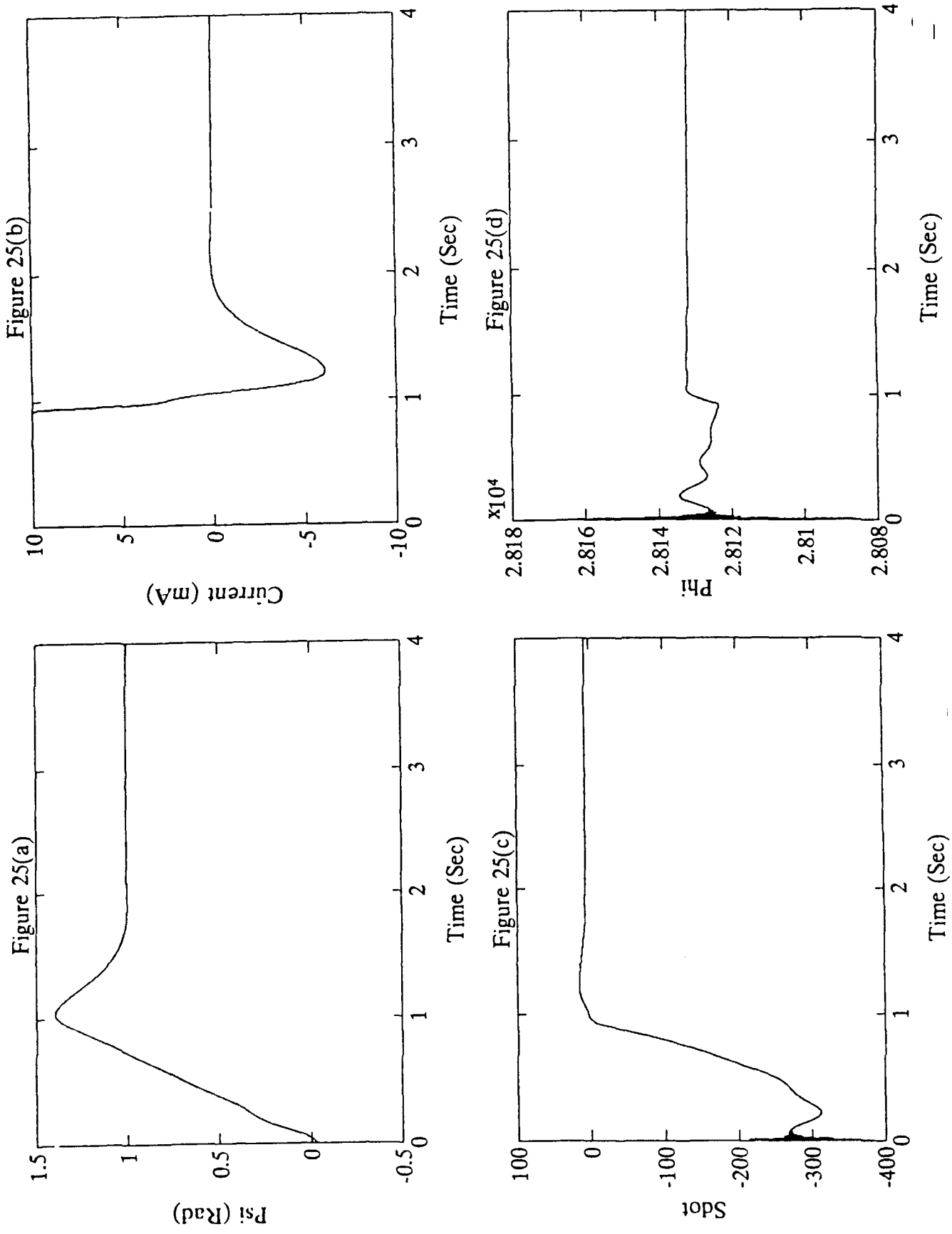

Figure 25(a)-(d) The Simulation Performance of Versatile Sliding Control
(ω_n = 1 rad/sec, z = 1, ζ_L = 10, Δt = 0.0002 (sec), payload = 0.00 kg)

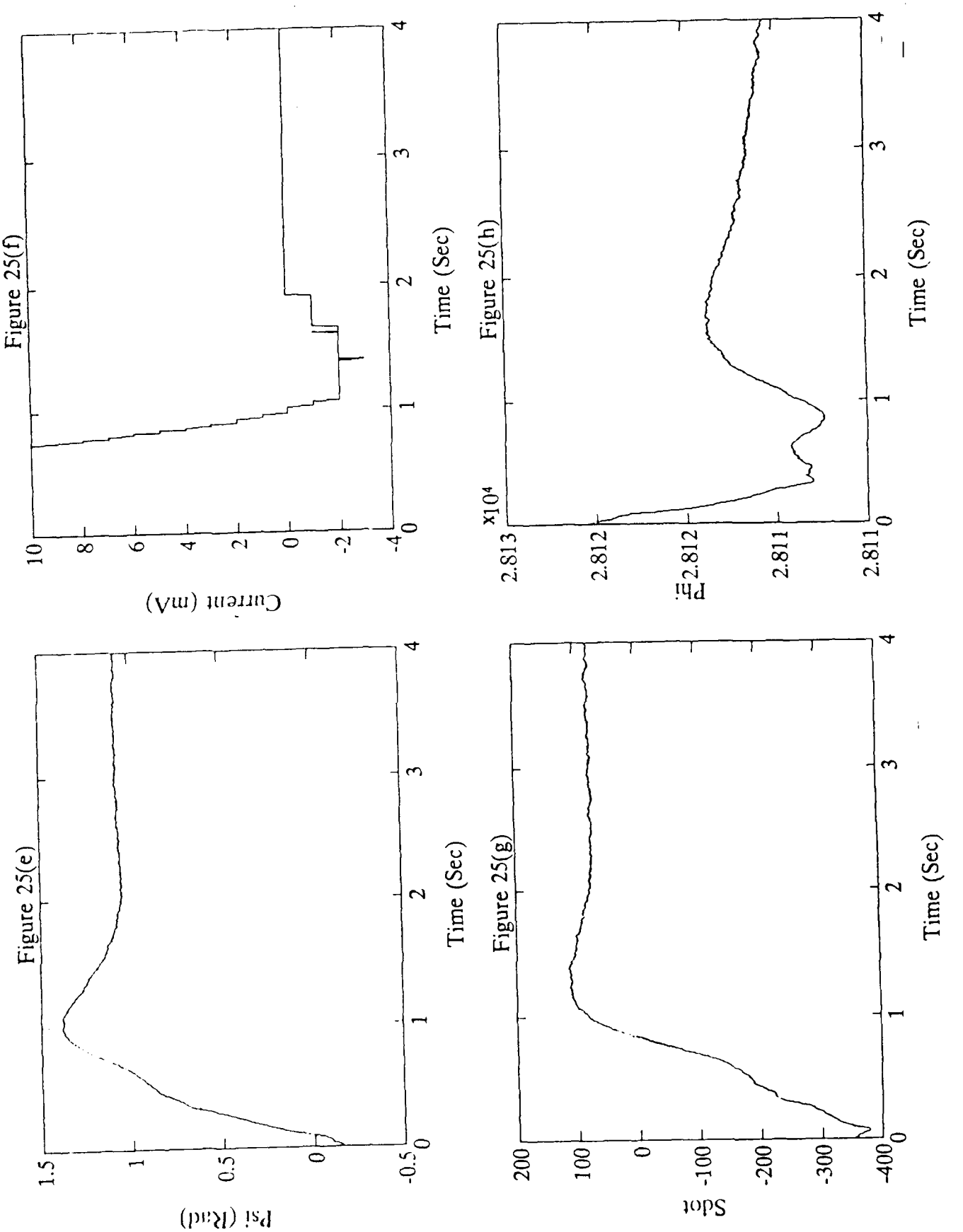

Figure 25(e)-(h) The Experiment Performance of Versatile Sliding Control
(ω_n = 1 rad/sec, z = 1, ζ_L = 10, payload = 0.00 kg)

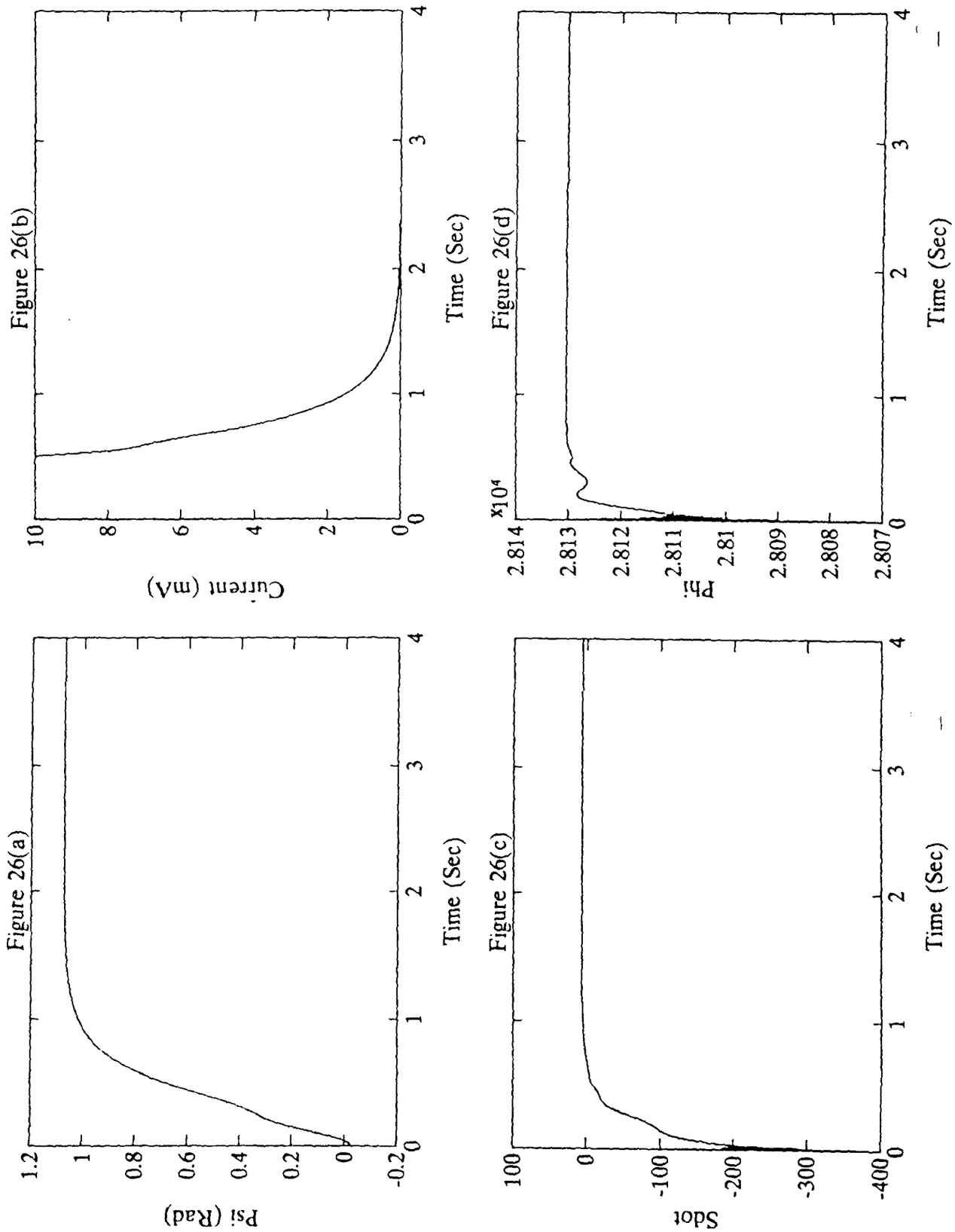

Figure 26(a)-(d) The Simulation Performance of Versatile Sliding Control
(ω_n = 1 rad/sec, z = 10, ζ_L = 10, Δt = 0.0002 (sec), payload = 0.00 kg)

Figure 26(e)-(h) The Experiment Performance of Versatile Sliding Control
(ω_n = 1 rad/sec, z = 10, ζ_L = 10, payload = 0.00 kg)

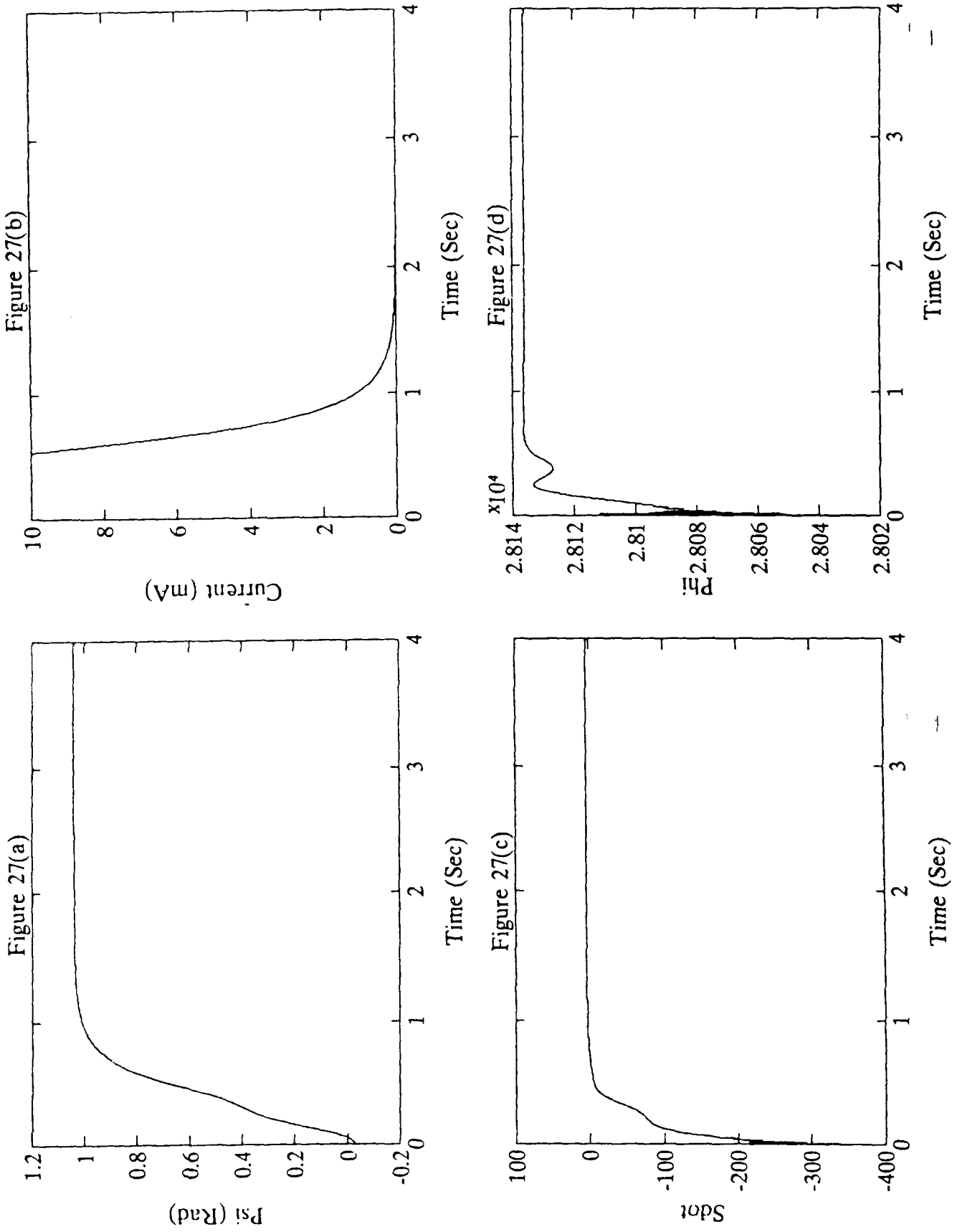

Figure 27(a)-(d) The Simulation Performance of Versatile Sliding Control
(ω_n = 1 rad/sec, z = 10, ζ_L = 10, Δt = 0.0002 (sec), payload = 0.85 kg)

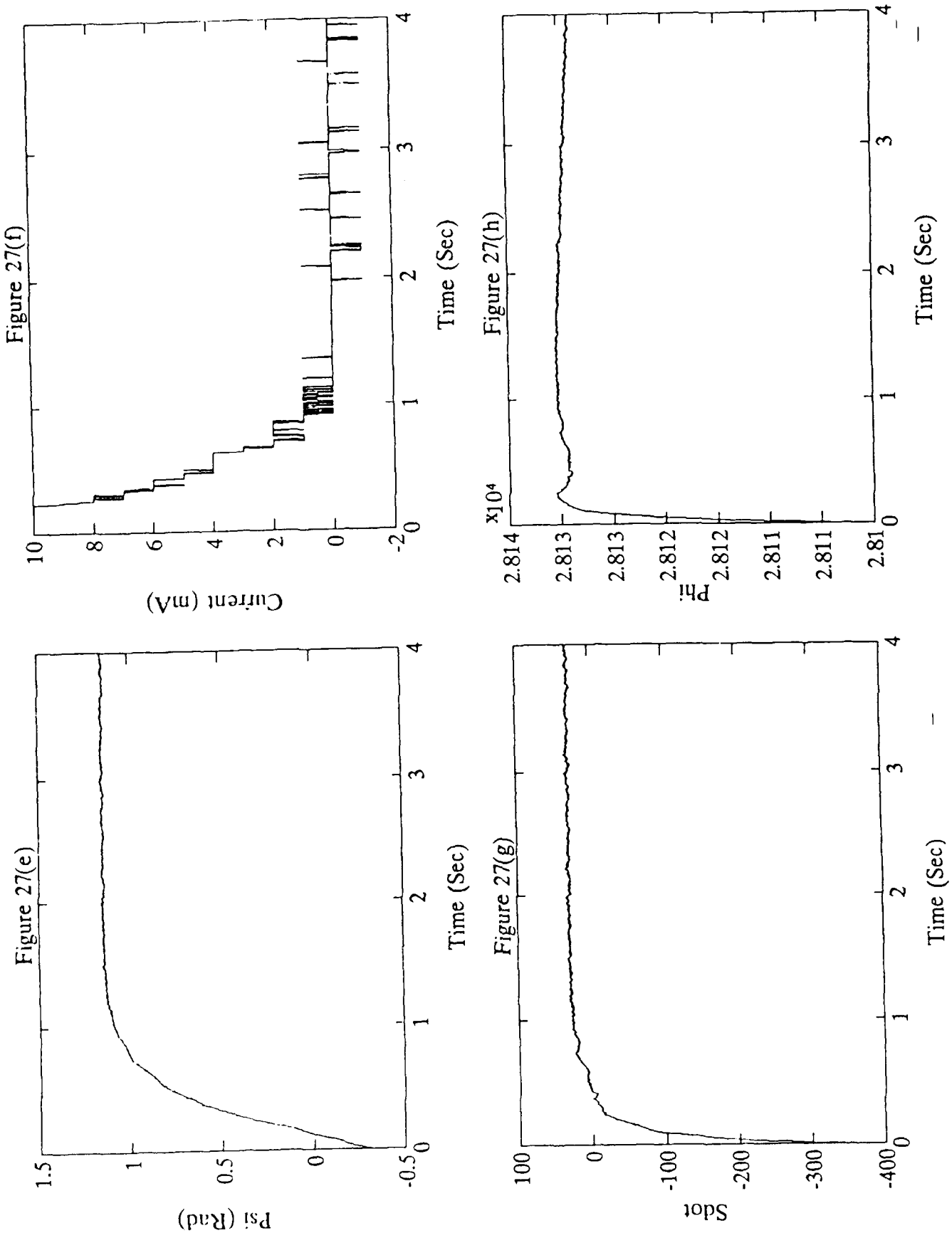

Figure 27(e)-(h) The Experiment Performance of Versatile Sliding Control
(ω_n = 1 rad/sec, z = 10, ζ_L = 10, payload = 0.85 kg)

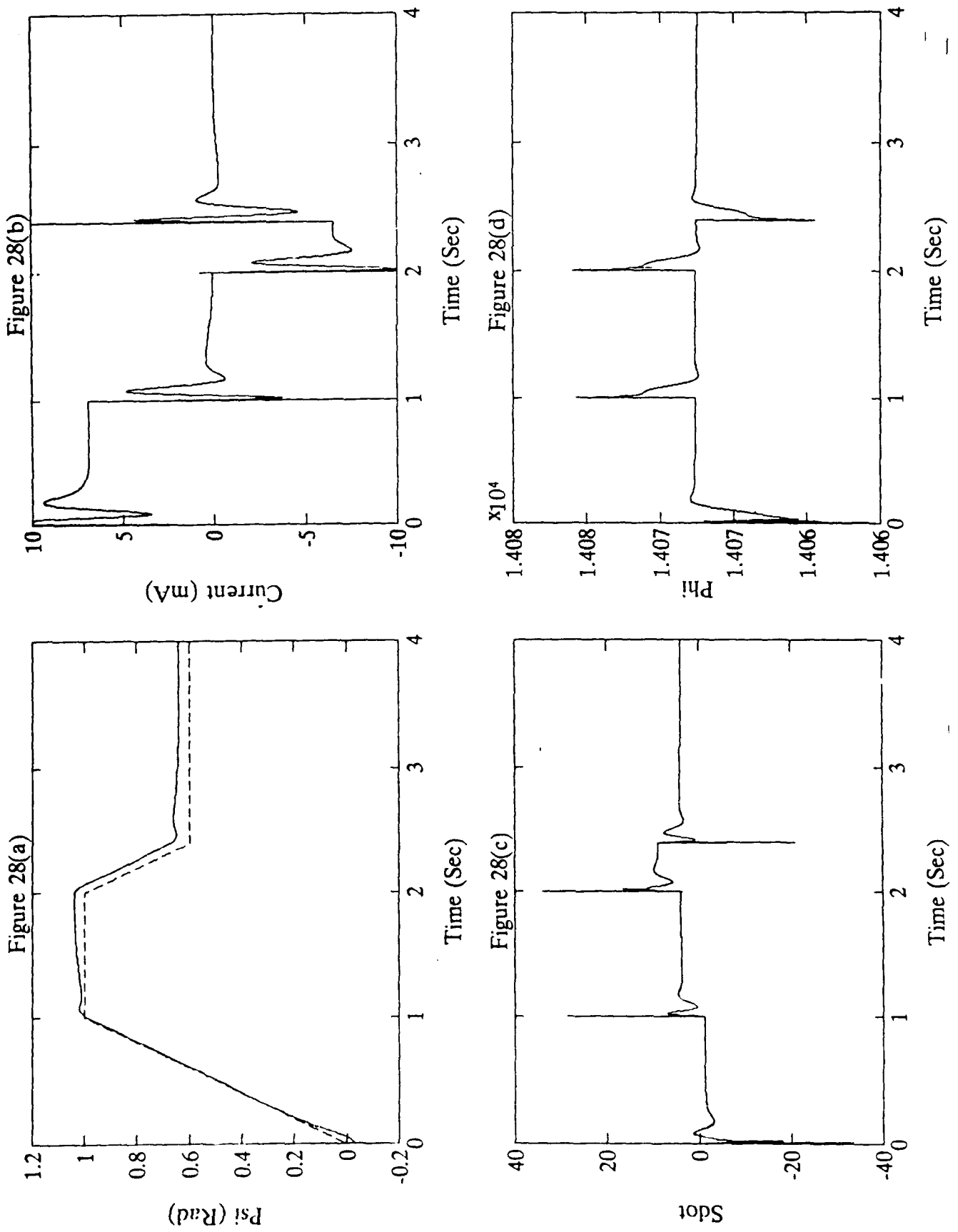

Figure 28(a)-(d) The Tracking Performance in Simulation of Versatile Sliding Control
(ω_n = 1 rad/sec, z = 10, ζ_L = 10, Δt = 0.0002 (sec), payload = 0.00 kg)

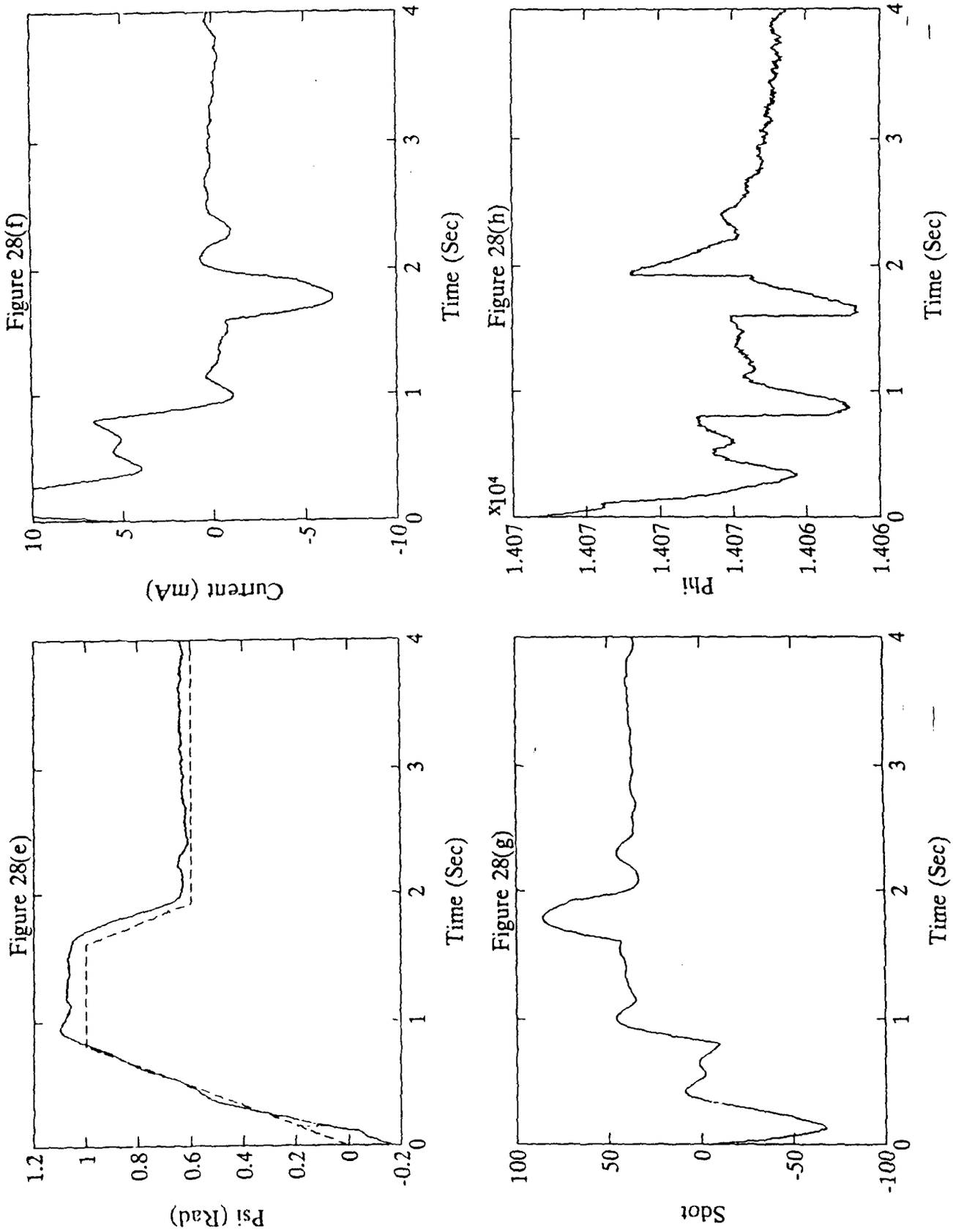

Figure 28(e)-(h) The Tracking Performance in Experiment of Versatile Sliding Control (ω_n = 1 rad/sec, z = 10, ζ_L = 10, payload = 0.00 kg)

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The purpose of doing this research was motivated by the need of using a flexible manipulator in the modern robotic application and of designing a simple robust control (sliding mode control) algorithm enabling a single-link flexible manipulator to perform accurate tracking under the environmental disturbances and uncertainties due to varying payload.

Because of the simple control structure of sliding mode control, the robust control system was implemented on a low-cost IBM-AT micro-computer for the flexible arm system. Unlike the high-speed switched sliding control, the predictor-corrector continuous control law was achieved by introducing a time-varying boundary layer. Three different forms of Lyapunov stability criterion (or sliding condition) were utilized not only to guarantee the stability of the control system but to provide different tuning capability.

In the sliding control with first-order sliding condition, the only control parameter λ_{max} not only provides the tightness of the boundary layer but also gives bandwidth of the S dynamics. The tighter the boundary layer, the faster the response speed and the smaller the steady-state error. However, the trade-off between tracking accuracy and the robustness to the unmodeled high-frequency dynamics exists. The integral error control will drive the steady-state error to zero.

The straight sliding control with first-order plus integral sliding condition provides control tuning parameters (ω_n and ζ_L) such that the rejection of uncertainties and tracking accuracy can

be achieved at the same time. The integral of S provides an ability to have the steady-state values of S zero for constant input to the S dynamics, and the steady-state error of the system will be driven to zero. The equivalent spring constant (ω_n^2) provides an rejection to the unmodeled high-frequency dynamics. The lower bound of damping ratio (ζ_L) provides a fine tune on the thickness of boundary layer. The thinner the boundary layer thickness, the faster the response speed and the smaller the steady-state error. However, from the experiment, even with continuous control, the small chattering occurred while the arm wanted to stand still. This phenomenon has not been clear.

The versatile sliding control algorithm provides a second-order sliding condition. The bandwidth of S dynamics was selected to be low so that the high-frequency unmodeled uncertainties will be rejected and the error dynamics will not be excited. The lower bound of damping and ζ in the S dynamics can then be fine tuning to tight the boundary layer so that the tracking accuracy can then be achieved. Therefore, the trade-off between the tracking accuracy and the robustness to the uncertainties has no longer existed. However, the same chattering in control input as in the straight sliding control occurred when the control was implemented.

In summary, despite the use of simplified model, the simple structure sliding control is indeed robustness to uncertainties due to varying payload. Also, by providing more tuning capability, the trade-off between tracking accuracy and the robust to the high-frequency dynamics was released.

B. RECOMMENDATIONS

The further works are recommended as follows :

(1) To develop a systematic procedure to tune controlling parameters.

(2) To clarify the small chattering occurred in the implementation while using straight and versatile sliding control.

(3) To develop the multiple-link control system using ERLS.

APPENDIX A

MATLAB SIMULATION PROGRAM

DEFINATIONS OF THE PARAMETERS :

```
*   lambda   = the max. bandwidth                h      = time interval
*   w        = tunning parmeter to tunne the max. bandwidth
*   zeta     = tunning parameter to provide more tunning capability
*   FT       = final time                        ml     = payload
*   R,G      = pole-placement of poles           speed  = sliding speed
*   Bhat     = B hat(uncrtainties)               DB     = DELTA B
*   fhat     = fhat(uncertainties)      Df       = DELTA f
*   L        = length of the arm        rho      = density of arm
*   A        = cross asection area of arm        E      = modulusof elasticity
*   I        = Area moment of inertia of flexible arm
*   Dv       = deformation matrix differentiation w.r.t. deflection
*   g        = gravitational acceleration vector
*   K        = servovalve sizing constant
*   Dm       = actuactor displacement
*   Vt       = total compress volumeincluding actuactor lines and chambers
*   Ps       = hydraulic supply pressure
*   Ctm      = total leaking coefficient
*   eff      = torque efficiency
*   betae    = effective bulk modulus
*   Ip       = moment of inertia of payload
*   Ir       = moment of inertia of
*   beta1    = mode shape coefficient
*   beta2    = mode shapew coefficient
*   C1       = computed mode shape coefficient
*   C2       = computed mode shape coefficient
*   n1m      = to evaluate the coeffieient and be used in fa
*   n1n1m    = to evaluate the coeffieient and be used in maa
*   lxn1m    = to evaluate the coefficient and be used in mab or mba
*   swwsm    = to evaluate the coefficient and be used in maa
*   klei     = to evaluate the coefficient and be used in kb
*   maa      = coefficient of large motion acceleration in large motion dynamics
*   mab      = coefficient of small motion acceleration in large motion dynamics
*   mba      = coefficient of large motion acceleration in small motion dynamics
*   mbb      = coefficient of smalle motion acceleration in small motion dynamics
*   kb       = coefficient of stiffness in small motion dynamics
*   fa       = right-hand-side of large motion dynamics
*   fb       = right-hand-side of small motion dynamics
*   theta    = large motion angle and thetaold represents previous one in
*              computation, thetanew represents the new one.
```

```
*       thetad  = velocity of large motion with the same expression as theta
*       thetadd         = acceleration of large motion with tha same expression as theta
*       v               = small motion and vold represents previous one in the computation
*                         and the vnew represents the new one.
*       vd              = velocity of small motion and has the same expression as in v.
*       vdd             = acceleration of small motion and has the same expression as in v
*       pl              = load hydraulic pressure drop
*       Ps              = hydraulic supply pressure
*       current         = the current input the electrohydraulic actuactor
*       psi             = total angle is the sum of large motion (theta) and small motion (v)
*       psiv            = velocity of total angle
*       psia            = acceleration of total angle
*       psir            = desired total angle
*       psivr           = the first time derivative of psir
*       psiar           = the second time derivative of psir
*       psijk           = the third time derivative of psir
*   Function expalnation
*       const1  = to perform the computation of constant in the program
*       timinvarm       = to perform the computation of time invariant terms
*       ic              = to perform the computation of initial condition
*       tracking        = to give the desired trajectory
*       sequenm         = to perform the sequential integration method
*       tippva          = to perform the transformation of total expression
*       ismco           = to compute the current using sliding control with first-order
*                         sliding condition
*       ismcst          = to compute the current using the straight sliding control
*       ismcva          = to compute the current using the versatile sliding control
*       hydralm         = to perform the dynamics of electrohydraulic actuactor
*       plotter = to perform the plot
*************************************************************************

% sliding control with first-order sliding condition algorithm
clear,clg
tra=input('Enter desired trajectory 1 or 2  ')
lambda=input('please enter lambda    ')

h=0.001;FT=10;t=0:h:FT;
ml=0;speed=10;R=[300;30;1];G=[1000;0;0];

Bhat=[0;0;1.1e4];DB=295;fhat=[0;0;-5.7732e4];Df=6.7117e4;
Ahat=[0 1 0;0 0 1;0 0 0];DA=0;

[L,rho,A,E,I,Dv,g,K,Dm,Vt,Ps,Ctm,eff,betae,Ip,Ir,beta1,beta2,C1,C2,B1,B2].....
=const1(ml);
```

```
% computation for the time invariant terms
[n1m,n1n1m,lxn1m,swwsm,k1ei,mab,mba,mbb]=timinvarm(rho,A,E,I,beta1,beta2.....
,C1,C2,B1,B2,Dv,Ip,L,ml);

% Initial Conditions
[thetaold,thetadold,thetaddold,vold,vdold,vddold,torqueactold,pl,current(1)....
,psi(1),psiv(1),psia(1)]=ic(Dm,eff,Ctm,Ps,K,L);

% desired trajectory
[psir,psivr,psiar,psijk]=tracking(t,tra);

for n=1:(length(t)-1)

n % sequential integration method
[thetanew,thetadnew,thetaddnew,vnew,vdnew,vddnew]=sequenm(thetaold,thetadold...
,thetaddold,vold,vdold,vddold,mab,mba,mbb,swwsm,n1n1m,lxn1m,n1m,k1ei,Ip.....
,Ir,Dv,ml,g,h,L,A,rho,torqueactold);

[psi(n+1),psiv(n+1),psia(n+1)]=tippva(thetanew,thetadnew,thetaddnew,vnew.....
,vdnew,vddnew,L);

% sliding mode control with first-order sliding condition
[current(n+1),s(n+1),intgerror,phi(n+1)]=ismco(Ahat,Bhat,fhat,DA,DB,Df,R,G....
,h,lambda,psi(n+1),psiv(n+1),psia(n+1),psir(n+1),psivr(n+1),psiar(n+1).....
,psijk(n+1),speed,intgerror);

% electriohydraulic actuactor
[plnew,torqueactnew]=hydralm(h,current(n+1),thetadnew,pl,K,Dm,Vt,Ps,Ctm.....
,eff,betae);
pl = plnew;

%
thetaold=thetanew;thetadold=thetadnew;thetaddold=thetaddnew;
vold=vnew;vdold=vdnew;vddold=vddnew;
torqueactold=torqueactnew;
end

% plotter -->
[ps,pphi,pcurrent,ppsi]=plotter(s,phi,current,psi)
*************************************************************************

% Straight Sliding Mode Control Algorithm
clear,clg
tra=input('Enter desired trajectory 1 or 2   ')
```

```
w=input('please enter w   ')
zeta=input('please enter zeta   ')

h=0.0002;FT=1.2;t=0:h:FT;ml=0;
R=[300;30;1];G=[1000;0;0];speed=10;

% Uncertainties
Bhat=[0;0;1.1e4];DB=295;
fhat=[0;0;-5.7732e4];Df=6.7114e4;
Ahat=[0 1 0;0 0 1;0 0 0];DA=0;

% Define constants
[L,rho,A,E,I,Dv,g,K,Dm,Vt,Ps,Ctm,eff,betae,Ip,Ir,beta1,beta2,C1,C2,B1,B2].....
=const1(ml);

% computation for some time invariant terms
[n1m,n1n1m,lxn1m,swwsm,k1ei,mab,mba,mbb]=timinvarm(rho,A,E,I,beta1,beta2.....
,C1,C2,B1,B2,Dv,Ip,L,ml);

% Initial conditions
[thetaold,thetadold,thetaddold,vold,vdold,vddold,torqueactold,pl.....
,current(1),psi(1),psiv(1),psia(1)]=ic(Dm,eff,Ctm,Ps,K,L);

% desired trajectory
[psir,psivr,psiar,psijk]=tracking(t,tra);

% Define initial values used in straight and intgral error sliding control
[intgerr,intgs,s(1),phi(1)]=smciv(R,G,psi(1),psiv(1),psia(1),psir,psivr.....
,psiar,h,w,zeta);

for n=1:(length(t)-1)
n
[thetanew,thetadnew,thetaddnew,vnew,vdnew,vddnew]=sequenm(thetaold.....
,thetadold,thetaddold,vold,vdold,vddold,mab,mba,mbb,swwsm,n1n1m,lxn1m,n1m.....
,k1ei,Ip,Ir,Dv,ml,g,h,L,A,rho,torqueactold);

[psi(n+1),psiv(n+1),psia(n+1)]=tippva(thetanew,thetadnew,thetaddnew.....
,vnew,vdnew,vddnew,L);

% straight sliding mode control
[current(n+1),s(n+1),intgs,intgerr,phi(n+1)]=ismcst(Ahat,Bhat,fhat,DA,DB.....
,Df,R,G,h,w,psi(n+1),psiv(n+1),psia(n+1),psir(n+1),psivr(n+1),psiar(n+1).....
,psijk(n+1),speed,intgs,intgerr,zeta);
```

```
% electrohydraulic actuactor
[plnew,torqueactnew]=hydralm(h,current(n+1),thetadnew,pl,K,Dm,Vt,Ps,Ctm.....
,eff,betae);
pl=plnew;
%
thetaold=thetanew;thetadold=thetadnew;thetaddold=thetaddnew;
vold=vnew;vdold=vdnew;vddold=vddnew;
torqueactold=torqueactnew;
end

% plotter -->
[ps,pphi,pcurrent,ppsi]=plotter(s,phi,current,psi)
**************************************************************************************

% Versatile Sliding Mode Control Algorithm
clear,clg
tra=input('Enter desired trajectory 1 or 2   ')
w=input('please enter w   ')
zo=input('please enter z   ')
zeta=input('please enter zeta   ')

h=0.0002;FT=1.2t=0:h:FT;ml=0;
speed=10;R=[300;30;1];G=[1000;0;0];

% Uncertainties
Bhat=[0;0;1.1e4];DB=295;
fhat=[0;0;-5.7732e4];Df=6.7117e4;
Ahat=[0 1 0;0 0 1;0 0 0];DA=0;
s=0;intgerr=0;

% Define constants
[L,rho,A,E,I,Dv,g,K,Dm,Vt,Ps,Ctm,eff,betae,Ip,Ir,beta1,beta2,C1,C2,B1,B2].....
=const1(ml);

% computation for some time invariant terms
[n1m,n1n1m,lxn1m,swwsm,k1ei,mab,mba,mbb]=timinvarm(rho,A,E,I,beta1,beta2.....
,C1,C2,B1,B2,Dv,Ip,L,ml);

% Initial conditions
[thetaold,thetadold,thetaddold,vold,vdold,vddold,torqueactold,pl,current(1).....
,psi(1),psiv(1),psia(1)]=ic(Dm,eff,Ctm,Ps,K,L);

% Desired trajectory
[psir,psivr,psiar,psijk]=tracking(t,tra);
```

```
for n=1:(length(t)-1)
n % sequential integration method
[thetanew,thetadnew,thetaddnew,vnew,vdnew,vddnew]=sequenm(thetaold.....
,thetadold,thetaddold,vold,vdold,vddold,mab,mba,mbb,swwsm,n1n1m,lxn1m.....
,n1m,k1ei,Ip,Ir,Dv,ml,g,h,L,A,rho,torqueactold);

[psi(n+1),psiv(n+1),psia(n+1)]=tippva(thetanew,thetadnew,thetaddnew.....
,vnew,vdnew,vddnew,L);

% Versatile sliding mode control
[current(n+1),sdot(n),s,intgerr,phi(n)]=ismcva(Ahat,Bhat,fhat,DA,DB,Df,R,G....
,psi(n+1),psiv(n+1),psia(n+1),psir(n+1),psivr(n+1),psiar(n+1),psijk(n+1).....
,speed,s,intgerr,h,w,zo,zeta);

% electrohydraulic actuator
[plnew,torqueactnew]=hydralm(h,current(n+1),thetadnew,pl,K,Dm,Vt,Ps.....
,Ctm,eff,betae);
pl=plnew;
%
thetaold=thetanew;thetadold=thetadnew;thetaddold=thetaddnew;
vold=vnew;vdold=vdnew;vddold=vddnew;
torqueactold=torqueactnew;
end

% plotter -->
[psdot,pphi,pcurrent,ppsi]=plotter(sdot,phi,current,psi)
*****************************************************************************

function [L,rho,A,E,I,Dv,g,K,Dm,Vt,Ps,Ctm,eff,betae,Ip,Ir,beta1,beta2,C1.....
,C2,B1,B2] = const1(ml)

rho = 7861.05;A = 6.17795e-04;
L = 0.9985;g = [0;0;-9.8066];
E = 2.0e11;I = 4.065e-10;
Dv= [0 0 0;0 0 0;1 0 0];     % first derivative of D w.r.t. v
Iyy = (1.3653333e-3)*ml;
Ixx= ((ml/0.4233)^2)*(7.5e-06)*ml;
Ip = [ml 0 0;0 Ixx 0;0 0 Iyy];     % Ip = moment of inertia of payload
mr = 9.00011451;Ir = [mr 0 0;0 0.02746713 0;0 0 0.02746713];
K  = 2.402963e-09;            % unit = ( m^4/(sec-mA-sqrt(N)) )
Dm = 6.2271e-05;              % unit = ( m^3/rad )
Vt = 3.05127e-04;             % unit = ( m^3 )
Ps = 1.37888e07;              % unit = ( N/m^2 )
Ctm = 3.7064772e-13;          % unit = ( m^5/N-sec )
```

```
betae = 6.90e08;              % unit = ( N/m^2 )
eff = 0.9;
beta1L = 1.875104069;beta2L = 4.694091133;
beta1 = beta1L/L;beta2 = beta2L/L;
C11 = sin(beta1L) + sinh(beta1L);
C12 = cos(beta1L) + cosh(beta1L);
C1 = C11/C12;
C21 = sin(beta2L) + sinh(beta2L);
C22 = cos(beta2L) + cosh(beta2L);
C2 = C21/C22;
D = 4*C1*beta2 - 4*C2*beta1;
B1 = 2*beta2/D;B2 = -2*beta1/D;
*********************************************************************************

function [alfa,delta,gama0,gama1,gama2,gama3,gama4,gama5] = const2(h)

alfa = 0.25;delta = 0.5;

gama0 = 1/(alfa*h^2);
gama1 = delta/(alfa*h);
gama2 = 1/(alfa*h);
gama3 = (1/(2*alfa))-1;
gama4 = (delta/alfa)-1;
gama5 = (h/2)*((delta/alfa)-2);
*********************************************************************************

function [alfa,delta,gama0,gama1,gama2,gama3,gama4,gama5,gama6,gama7]=const3(h)

alfa = 0.25;delta = 0.5;

gama0 = 1/(alfa*h^2);
gama1 = delta/(alfa*h);
gama2 = 1/(alfa*h);
gama3 = (1/(2*alfa))-1;
gama4 = (delta/alfa)-1;
gama5 = (h/2)*((delta/alfa)-2);
gama6 = h*(1-delta);
gama7 = delta*h;
*********************************************************************************

function [n1m,n1n1m,lxn1m,swwsm,k1ei,mmab,mmba,mmbb]=timinvarm(rho,A,E,I.....
,beta1,beta2,C1,C2,B1,B2,Dv,Ip,L,ml)

swwsm = simps('swws',0,0,0,0,0,0)*rho*A;
```

```
n1m = simps('n1',beta1,beta2,C1,C2,B1,B2)*rho*A;
k1ei = simps('k11',beta1,beta2,C1,C2,B1,B2)*E*I;
n1n1m = simps('n1n1',beta1,beta2,C1,C2,B1,B2)*rho*A;
lxn1m = simps('lxn1',beta1,beta2,C1,C2,B1,B2)*rho*A;
[D,Dd,w,wd,wdd,Ad,Add] = usualm(-pi/3,0,7,L);
mmba = mba(D,w,wd,Dv,Ip,lxn1m);
mmbb = mbb(w,Dv,Ip,n1n1m,ml);
mmab = mab(D,w,wd,Dv,Ip,lxn1m,ml,L);
*********************************************************************************

function [theta1,thetad1,thetadd1,v1,vd1,vdd1,torqueact1,pl,current1,psi1.....
,psiv1,psia1]=ic(Dm,eff,Ctm,Ps,K,L)

theta1=0;thetad1=0;thetadd1=0;
v1=-0.03111;%v1=-0.06111;%v1=-0.146064149;%v1=-0.23155;
vd1=0;vdd1=0;

torqueact1= 44.5549095;pl=torqueact1/(Dm*eff);
Q=Ctm*pl;I0=Q/(K*sqrt(Ps-pl));Imax=10;
ifrac=0.5*(Imax-I0);current1=I0+ifrac;
*********************************************************************************

% Trajectory design function
%
function [psid,psivd,psiad,psijerk]=tracking(t,n)

if (n==1)
   psid=ones(t);psivd=zeros(t);psiad=zeros(t);psijerk=zeros(t);
elseif(n==2)
   psijerk=0;psiad=0;
   for n=1:length(t)
      if (t(n)<1)
         psid(n)=t(n);psivd(n)=1;
      elseif (1<=t(n) & t(n)<2)
         psid(n)=1;psivd(n)=0;
      end
      if (2<=t(n) & t(n)<2.4)
         psid(n)=3-t(n);psivd(n)=-1;
      elseif (t(n)>=2.4)
         psid(n)=0.6;psivd(n)=0;
      end
   end
end
*********************************************************************************
```

```
%
%  Sequential integration method is used to do numerical iteration.
%

function [thetanew,thetadnew,thetaddnew,vnew,vdnew,vddnew] = sequen(theta.....
,thetad,thetadd,v,vd,vdd,mab,mba,mbb,swwsm,n1n1m,lxn1m,n1m,k1ei,Ip,Ir,Dv.....
,ml,g,h,L,A,rho,torque)

[alfa,delta,gama0,gama1,gama2,gama3,gama4,gama5] = const2(h);

% following steps are solving for small-motion(vnew, vdnew, & vddnew)
%  (1) original coefficients computation ( large and small-motion ) at time t
[D,Dd,w,wd,wdd,Ad,Add] = usualm(theta,v,vd,L);
Kb  = kb(thetad,k1ei,n1n1m);
Fb  = fb(thetad,w,wd,D,Dd,n1m,Dv,Ip,g,ml);
Maa = maa(v,wd,Ad,D,Ip,Ir,swwsm,n1n1m,A,rho);
Fa  = fa(v,vd,thetad,wd,wdd,Ad,Add,D,Dd,Ip,Ir,n1m,n1n1m,g,ml,torque);

% (2) modify small-motion coefficients ( at time t ) to avoid secular terms
Mbs = mbb - mba*(1/Maa)*mab;
Fbs = Fb - mba*(1/Maa)*Fa;
omega = (1/(2*pi))*sqrt(Kb/Mbs);
C = 0.2*(sqrt(Kb/Mbs));

%  (3) fast (small) motion solver --> vnew, vdnew, and vddnew
Kbi = Kb + gama0*(Mbs) + gama1*C;
Fbi=Fbs+Mbs*(gama0*v+gama2*vd+gama3*vdd)+C*(gama1*v+gama4*vd+gama5*vdd);
vnew = (inv(Kbi))*Fbi;
vdnew=vd+(1-delta)*h*vdd+gama1*(vnew-v)-((delta/alfa)*vd)-gama3*delta*h*vdd;
vddnew = gama0*(vnew-v) - gama2*vd - gama3*vdd;

% using predictor and corrector scheme to find large-motion
% (1) using predictor and new etas' to evaluate large-motion coefficients
thetap = theta + thetad*h + (0.5-alfa)*(h*h)*(thetadd);
thetadp = thetad + (1-delta)*h*thetadd;
[D,Dd,w,wd,wdd,Ad,Add] = usualm(thetap,vnew,vdnew,L);
Maa = maa(vnew,wd,Ad,D,Ip,Ir,swwsm,n1n1m,A,rho);
Fa  = fa(vnew,vdnew,thetadp,wd,wdd,Ad,Add,D,Dd,Ip,Ir,n1m,n1n1m,g,ml,torque);

% (2) solve for thetaddnew
Fae = Fa - mab*vddnew;
thetaddnew = Fae/(Maa);

% (3) find correctors using thetaddnew
```

```
thetac = alfa*(h*h)*thetaddnew;
thetadc = delta*h*thetaddnew;

% (4) solve for thetanew and thetadnew
thetadnew = thetadp + thetadc;
thetanew = thetap + thetac;
*****************************************************************************

function [x,xr,xrd]=tippva(theta,thetad,thetadd,v,vd,vdd,psid,psivd,psiad.....
,psij,L)

psi = theta + (v/L);
psiv = thetad + (vd/L);
psia = thetadd + (vdd/L);
x=[psi;psiv;psia];
xr=[psid;psivd;psiad];
xrd=[psivd;psiad;psij];
*****************************************************************************

function [current,s,intgerror,phi]=ismco(Ahat,Bhat,fhat,DA,DB,Df,R,G,h,lambda.
,psi,psiv,psia,psir,psivr,psiar,psijk,speed,intgerror);

x=[psi;psiv;psia];xr=[psir;psivr;psiar];xrd=[psivr;psiar;psijk];
error = x - xr;
intgerror=intgerror+error*h;
RtB=inv(R'*Bhat);RtBn=norm(RtB,2);
s=R'*error+G'*intgerror;
RtBsgn=RtB*sign(s);RtBsn=norm(RtBsgn,2);

% norminal value
Ihat = R'*(xrd-fhat-Ahat*x)-G'*error;
Ihatn= norm(Ihat,2);

box1=RtB*Ihat;
box1n=norm(box1,2);

% k
Rn=norm(R,2);
k=((Rn)*(Df+DB*box1n)+speed)/(1-(DB*RtBsn));
phi=(k/lambda)*(1+DB*RtBn);

std=s/phi;
if( abs(s)>=phi )
   if( std>=0 )
```

```
        sat=1;
    else
        sat=-1;
    end
elseif( abs(s)<phi )
    sat=std;
end
current=(RtB)*(Ihat-k*sat);
if( current >= 10)
  current=10;
end
if( current <= 10)
  current=-10;
end
*****************************************************************************

function [current,s,intgs,intgerr,phi]=ismcst(Ahat,Bhat,fhat,DA,DB,Df,R,G.....
,h,w,psi,psiv,psia,psir,psivr,psiar,psijk,speed,intgs,intgerr,zeta);

x=[psi;psiv;psia];xr=[psir;psivr;psiar];xrd=[psivr;psiar;psijk];
error=x-xr;s=R'*error;
intgs=intgs+s*h;
RtB=inv(R'*Bhat);RtBn=norm(RtB,2);
RtBsgn=RtB*sign(s);RtBsn=norm(RtBsgn,2);

% norminal value
Ihat=R'*(xrd-fhat-Ahat*x)-(w^2)*intgs;
Ihatn=norm(Ihat,2);

box1=RtB*Ihat;
box1n=norm(box1,2);

% k
Rn=norm(R,2);
k=((Rn)*(Df+DB*box1n)+speed)/(1-(DB*RtBsn));

% tuning parameters
lambda=(sqrt(1+DB*RtBn))*w;
phi=(k/(2*zeta*lambda))*(DB*RtBn);

std=s/phi;
if( abs(s)>=phi )
  if( std>=0 )
      sat=1;
```

```
   else
      sat=-1;
   end
elseif( abs(s)<phi )
    sat=std;
end
% (3) current
current=RtB*(Ihat-k*sat);
if( current >= 10)
   current=10;
end
if( current <=-10)
   current=-10;
end
*****************************************************************************

function [current,sdot,s,intgerr,phi]=ismcva(Ahat,Bhat,fhat,DA,DB,Df,R,G.....
,psi,psiv,psia,psir,psivr,psiar,psijk,speed,s,intgerr,h,w,zo,zeta);

x=[psi;psiv;psia];xr=[psir;psivr;psiar];xrd=[psivr;psiar;psijk];
error=x-xr;

intgerr=intgerr+error*h;
sdot=R'*error+G'*intgerr-zo*s;
s=s+sdot*h;
RtB=inv(R'*Bhat);RtBn=norm(RtB,2);
RtBsgn=RtB*sign(sdot);RtBsn=norm(RtBsgn,2);

% norminal value
Ihat=R'*(xrd-fhat-Ahat*x)-(w^2)*s-G'*error+zo*sdot;
Ihatn=norm(Ihat,2);

box1=RtB*Ihat;
box1n=norm(box1,2);

% k
Rn=norm(R,2);
k=((Rn)*(Df+DB*box1n)+speed)/(1-(DB*RtBsn));

% tuning parameters
lambda=(sqrt(1+DB*RtBn))*w;
phi=(k*(DB*RtBn))/((2*zeta*lambda)+(DB*RtB*zo));

% sat(sdot) FUNCTION
```

```
std=sdot/phi;
if( abs(sdot)>=phi )
  if( std>=0 )
     sat=1;
  else
     sat=-1;
  end
elseif( abs(sdot)<phi )
   sat=std;
end
% CURRENT
current=RtB*(Ihat-k*sat);
if( current >= 10)
  current=10;
end
if( current <=-10)
  current=-10;
end
*****************************************************************************

function [Plnew,torque]=hydralm(h,current,thetad,Pl,K,Dm,Vt,Ps,Ctm,eff,betae)
%
% A fourth-order Runge-Kutta method is used to obtain the numerical solution
% In this case, the only variable will be Pl
%
Pv = 'Ps - (Pl+x)';
Pld = '(4*betae/Vt) * (K*current*(sqrt(eval(Pv))) - Dm*thetad - Ctm*(Pl+x) )';

% To evaluate k1, let x = 0
x = 0;
if (eval(Pv)) < 0
   Pv = '0';
end
k1 = h*eval(Pld);
% To evaluate k2, let x = k1/2
x = k1/2;
if (eval(Pv)) < 0
   Pv = '0';
end
k2 = h*eval(Pld);
% To evaluate k3, let x = k2/2
x = k2/2;
if (eval(Pv)) < 0
   Pv = '0';
```

```
end
k3 = h*eval(Pld);
% To evaluate k4, let x = k3
x = k3;
if (eval(Pv)) < 0
   Pv = '0';
end
k4 = h*eval(Pld);
Plnew = Pl + (1/6)*(k1 + 2*k2 + 2*k3 + k4);
if Plnew > Ps
  Plnew = Ps;
end
if Plnew < -Ps
  Plnew = -Ps;
end
torque = eff*Plnew*Dm;
*****************************************************************************

function [ps,pphi,pcurrent,ppsi]=plotter(s,phi,current,psi,t)
subplot(221),plot(t,s),xlabel('Time(sec)'),ylabel('S')
subplot(221),plot(t,phi),xlabel('Time(sec)'),ylabel('Phi')
subplot(221),plot(t,current),xlabel('Time(sec)'),ylabel('Current (mA)')
subplot(221),plot(t,psi),xlabel('Time(sec)'),ylabel('Psi (Rad)')
*****************************************************************************

function [D,Dd,w,wd,wdd,Ad,Add]=usualm(tha,v,vd,L)

D=[1 0 0;0 1 0;v 0 1];Dd=[0 0 0;0 0 0;vd 0 0];
w=[1 0 0;L*cos(tha) cos(tha) -sin(tha);L*sin(tha) sin(tha) cos(tha)];
wd=[0 0 0;-L*sin(tha) -sin(tha) -cos(tha);L*cos(tha) cos(tha) -sin(tha)];
wdd=[0 0 0;-L*cos(tha) -cos(tha) sin(tha);-L*sin(tha) -sin(tha) -cos(tha)];
Ad=[0 0 0;0 -sin(tha) -cos(tha);0 cos(tha) -sin(tha)];
Add=[0 0 0;0 -cos(tha) sin(tha);0 -sin(tha) -cos(tha)];
*****************************************************************************

function ans = fa(v,vd,thetad,wd,wdd,Ad,Add,D,Dd,Ip,Ir,n1m,n1n1m,g,ml,torque)
%
%  evaluation of right hand side of the large-motion equation
%
% (1)   -2*thetad*etad'*{integration [N(x)'*W,'*W,*N(x)] dm}*eta
%       where {integration [N(x)'*W,'*W,*N(x)]dm}=[nwwn1 nwwn12;nwwn12 nwwn2]

x = -2*thetad*vd*n1n1m*v;
```

```
% (2.1)  {integration[s'] dm}*W,'*g
%       where {integration[s'] dm} = s  <--  1st moment of inertia of link

s = [4.856519 -2.428258693 0];

y1 = s*wd'*g;

% (2.2)  eta'*{integration[N'] dm}*W,'*g
%       where {integration[N'] dm} = n

n = [0 0 n1m];

y2 = v*n*wd'*g;

% (2.3)  {integration[Sp']dm}*D'*W,'*g
%       where {integration[Sp'] dm} = Sp <-- 1st moment of inertia of payload

Sp = [ml 0 0];

y3 = Sp*D'*wd'*g;

y = y1+y2+y3;

%  (3) evaluation "trace" terms of right hand side of large-motion equation

%  trace [2*wd*D*Ip*Dd'*wd']*thetad

z11 = -2*thetad*wd*D*Ip*Dd'*wd';

z = trace(z11);

ans = x+y+z+torque;
*************************************************************************

function ans = fb(thetad,w,wd,D,Dd,n1m,Dv,Ip,g,ml)
%
%  evaluante the right hand side of the small-motion equation
%
% (1)  {Integration [N'] dm}*W'*g

n = [0 0 n1m];

u = n*w'*g;
```

```
% (2)
%    [ {integration [Sp'] dm}*Dv'*w'*g    {integration [Sp'] dm}*Dphi'*w'*g ]
%    where {integration [Sp'] dm} = Sp

Sp = [ml 0 0];

x = Sp*Dv'*w'*g;

% (3)
%  evaluate all the "trace" terms of right hand side of small-motion equation
%
%     (3.1)  trace [wdd*D*Ip*Dv'*w'*(thetad^2))+2*wd*Dd*Ip*Dv'*w']

y11 = wd*Dv*Ip*D'*wd'*(thetad^2);
y22 = 2*wd*Dv*Ip*Dd'*w'*(thetad);
y = trace(y11+y22);

ans = u+x+y;          % sum of the right hand side of small-motion equation
*************************************************************************************

function ans = maa(v,wd,Ad,D,Ip,Ir,swwsm,n1n1m,A,rho)
%                                                                    %
%  coefficient computation of "thetadd" of large-motion equation          %
%                                                                    %
% (1) {integration [s'*wd'*wd*s] dm} = swwsm
x = swwsm;

% (2)  eta'*{integration [N'*wd'*wd*N] dm}*eta
y  = v'*n1n1m*v;        % 2nd-order value should be small

%  The following is used to evaluate all the "trace" terms in the coef11
%  (2.1) Trace [ W,*D*Ip*D'*W,' ]
u1  = trace(wd*D*Ip*D'*wd');

%  (2.2) Trace [ A,*Ir*A, ]
u2 = trace(Ad*Ir*Ad');
z = u1+u2;

ans = x+y+z;
*************************************************************************************

function ans = mab(D,w,wd,Dv,Ip,lxn1m,ml,L)
%
%  coefficient computation of "etadd" of large-motion equation
```

```
%
% (1) {integration [s'*wd'*w*N] dm} = [swwn1 swwn2]
x = lxn1m;

% (2) [trace (W,*D*Ip*(Dv)'*W')   trace (W,*D*Ip*(Dphi)'*W')]
y = trace(wd*D*Ip*Dv'*w');

ans = x+y;
*****************************************************************************

function ans = mba(D,w,wd,Dv,Ip,lxn1m)
%
% coefficient computation of "etadd" of large-motion equation
%
% (1) {integration [s'*wd'*w*N] dm} = [swwn1 swwn2]
x = lxn1m;

% (2) [trace (W,*D*Ip*(Dv)'*W')   trace (W,*D*Ip*(Dphi)'*W')]
y = trace(wd*D*Ip*Dv'*w');

ans = x+y;
*****************************************************************************

function ans = mbb(w,Dv,Ip,n1n1m,ml)
%
% coefficient computation of "etadd" of small-motion equation
%
% (1) {integration [N'*w*w*N] dm}= [nwwn1 nwwn12;nwwn12 nwwn2]
x = n1n1m;

% (2) [ Tr(W*Dv*Ip*Dv'*W') 0;0 Tr(W*(Dphi)*Ip*(Dphi)'*W') ]
y = trace(w*Dv*Ip*Dv'*w');

ans = x+y;
*****************************************************************************

function ans = kb(thetad,klei,n1n1m)
%
% coefficient computation of "eta" of small-motion equation
%
% (1)  ((E*I)/2)*{integartion [sigma+sigma'] dx} = x
x = klei;

% (2)  {integration [N'*wd'*wd*N] dm}*eta*(thetad^2)
```

```
%       where {integration [N'*wd'*wd*N] dm} = [nwwn1 nwwn12;nwwn12 nwwn2]
y = -(thetad^2)*n1n1m;

ans=x+y;
*****************************************************************************

function ans = n1(x,beta1,beta2,C1,C2,B1,B2)
%
% natural-mode shape functions N1(x) evaluation
%
xi1 = C1*(cos(beta1*x)+cosh(beta1*x)) + (sin(beta1*x)+sinh(beta1*x));
xi2 = C2*(cos(beta2*x)+cosh(beta2*x)) + (sin(beta2*x)+sinh(beta2*x));
ans = B1*xi1 + B2*xi2;
*****************************************************************************

function ans = n1n1(x,beta1,beta2,C1,C2,B1,B2)
%
xi1 = C1*(cos(beta1*x)+cosh(beta1*x)) + (sin(beta1*x)+sinh(beta1*x));
xi2 = C2*(cos(beta2*x)+cosh(beta2*x)) + (sin(beta2*x)+sinh(beta2*x));
N1 = B1*xi1 + B2*xi2;
%  'nwwn1' is to evaluate N1^2 and to be used in "coef11.m"
%  An evaluation of [ N1^2 N1*N2 ; N1*N2 N2^2 ] which comes from multiplic-
%  ation of [ N(x)'*W,'*W,*N(x) ]
ans = N1.^2;
*****************************************************************************

function ans = lxn1(x,beta1,beta2,C1,C2,B1,B2)

xi1 = C1*(cos(beta1*x)+cosh(beta1*x)) + (sin(beta1*x)+sinh(beta1*x));

xi2 = C2*(cos(beta2*x)+cosh(beta2*x)) + (sin(beta2*x)+sinh(beta2*x));

N1 = B1*xi1 + B2*xi2;

ans = 0.9985*N1 + x.*N1;
*****************************************************************************

function y = swws(x,beta1,beta2,C1,C2,B1,B2)
%
%  To be used in "coef11.m" and evaluate integral [S'*W,'*W,*S]dx
%  After multiplication [ S'*W,'*W,*S ] = (L+x)^2
%  where W  represents 2-D homogeneous transform matrix
%        W, represents first derivative w.r.t. theta
%        S  represents the local position along the link [1 x 0]'
```

```
%       L  length of the link = 0.9985 m
%

y = (0.9985+x).^2;
*************************************************************************************

function ans = k11(x,beta1,beta2,C1,C2,B1,B2)

xi1dd=(beta1*beta1)*(C1*(-cos(beta1*x)+cosh(beta1*x))+(-sin(beta1*x).....
+sinh(beta1*x)));
xi2dd=(beta2*beta2)*(C2*(-cos(beta2*x)+cosh(beta2*x))+(-sin(beta2*x).....
+sinh(beta2*x)));
% Note: N1dd = second derivative of N1(x) w.r.t. x
%       N2dd = second derivative of N2(x) w.r.t. x
N1dd = B1*xi1dd + B2*xi2dd;
ans = N1dd.^2;
```

APPENDIX B

FORTRAN IMPLEMENTATION PROGRAMS

```
C First-Order Slid'ng Condition + An Integral Error Control
$INCLUDE:'ATLDEFS.FOR'
$INCLUDE:'ATLERRS.FOR'
      PARAMETER (NUM=1750)
      INTEGER*2 ADGAINS(16),ADCHAN(16),ICONFIG(16),BASEADR,CDEVID,
     +CDEVF,LG,SCAN,DAVAL1,DAVAL2,Y,STATUS
      INTEGER*2 FH,FM,FS,FSS
      REAL BHAT,FHAT,L,PI,K,R(3),G,IHAT,INTGERR,LAMBDA
      REAL PSID(NUM),PSIVD(NUM),PSIJERK(NUM),S(NUM),CURRENT(NUM)
      REAL PSI(0:NUM),PHI(NUM)
      COMMON/CONFIG/ICONFIG
      EQUIVALENCE
      (ICONFIG(KCBASEADR),BASEADR),(ICONFIG(KCDEVID),CDEVID)
     +,(ICONFIG(KCDEVFLAGS),CDEVFLG),(ICONFIG(KCSCAN),SCAN),(ICONFIG(KCC
     +HANNELS),CHAN)
C
      DATA R,G/300.0,30.0,1.0,3000.0/,SPEED,L/10.0,0.9985/
      DATA SMALFIL,INTGERR/2*0.0/,RATE,FREQ/2000.0,200.0/
      DATA PSIJERK/NUM*0.0/,PSIVOD,D2OLD,VOLD/3*0.0/
C UNCERTAINIES
      DATA BHAT,DB/1.1E4,295.0/,FHAT,DF/-5.7732E4,6.7114E4/
C
      PRINT *,'Please Enter Lambda for S dynamics, and Payload'
      READ *,LAMBDA,PAYLOAD
      PRINT *,'Please Enter Desired Trajectory "1",or "2" '
      READ *,IANS
C DESIRED TRAJECTORY
      IF(IANS .EQ. 1) THEN
        DO 100 M=1,NUM
         TIME=REAL(M)/FREQ
          IF (TIME .LE. 1.0) THEN
             PSID(M)=TIME
             PSIVD(M)=1.0
          ELSE IF((TIME .GE. 1.0) .AND. (TIME .LE. 2.0)) THEN
             PSID(M)=1.0
             PSIVD(M)=0.0
          END IF
          IF((TIME .GT. 2.0) .AND. (TIME .LE. 2.4)) THEN
             PSID(M)=3.0-TIME
             PSIVD(M)=-1.0
```

```
          ELSE IF(TIME .GE. 2.4) THEN
             PSID(M)=0.6
             PSIVD(M)=0.0
          END IF
100     CONTINUE
      ELSE IF(IANS .EQ. 2) THEN
        DO 500 M=1,NUM
         PSID(M)=1.0
         PSIVD(M)=0.0
         PSIJERK(M)=0.0
500     CONTINUE
      END IF
C
      M=0
      PI=4*ATAN(1.0)
      PAR=PI/1054.0
      R1B=1.0/(R(3)*BHAT)
      R1NOM=SQRT(R(1)**2+R(2)**2+R(3)**2)
      OPEN(UNIT=15,FILE='C:\ATLAB\TMP\CTRL1.M',STATUS='NEW')
C INITIALIZE DATA ACQUSITION BOARD
      STATUS=ALINIT()
      STATUS=ALSB(1)
      STATUS=ALSF(RATE)
      STATUS=ALRSET()
      STATUS=ALGC(ICONFIG)
      STATUS=ALDV(0,Y)
C
      WRITE(*,*)' Are you ready (Ctrl-C) ? '
      WRITE(*,*)' When you are ready to go, press "1" and "return" '
      READ *,IANS1
      IF(IANS1 .EQ. 1) GOTO 7
C BEGIN SIGNAL PROCESSING
  7  CALL GETTIM(IH,IM,IS,ISS)
  5  STATUS=ALAV(1,1,DAVAL1)
      STATUS=ALAV(2,1,DAVAL2)
      D1=REAL(DAVAL1-2048)
      D2=REAL(DAVAL2-2048)
C FILTERED VERSION OF SMALL MOTION SIGNAL COMING OUT FROM STRAIN
INDICATOR
      SMALFIL=0.94175*SMALFIL+0.029129*(D2OLD+D2)
C CALIBRATED EQUATION BETWEEN DISPLACEMENT(cm) AND SMALFIL
      SMALL=-0.3*(SMALFIL+15)
      SMANG=SMALL/(100.0*L)
C LARGE MOTION CALIBRATED EQUATION
```

```
      THETA=PAR*(D1-30.0)
C TOTAL ANGLE
      PSI(M)=THETA+SMANG
C CONTROLLER DESIGN
      IF(M .EQ. 0) GOTO 1
      POSERR=PSI(M)-PSID(M)
      INTGERR=POSERR/FREQ + INTGERR
C FILTERED VERSION OF TOTAL VELOCITY, PSIV(M)
      VNEW=(PSI(M)-PSI(M-1))*FREQ
      PSIV=0.980198*PSIVOD+0.00990099*(VOLD+VNEW)
      VELERR=PSIV-PSIVD(M)
      S(M)=R(1)*POSERR+R(2)*VELERR+G*INTGERR
C NORMINAL VALUE
      IHAT=R(1)*(-VELERR)+R(3)*(PSIJERK(M)-FHAT)-G*POSERR
C TO FIND K
      K=(R1NOM*(DF+DB*(R1B*IHAT))+SPEED)/(1.0-(DB*R1B))
      PHI(M)=(K/LAMBDA)*(1.0+DB*R1B)
      STD=S(M)/PHI(M)
      IF(ABS(S(M)) .LT. PHI(M)) SAT=STD
      IF(ABS(S(M)) .GE. PHI(M)) THEN
        IF(STD .GE. 0.0) SAT=1.0
        IF(STD .LT. 0.0) SAT=-1.0
      ENDIF
      CURRENT(M)=R1B*((IHAT - K*SAT)/1000.0)
      IF (CURRENT(M) .GE. 10.0E-03) CURRENT(M)=10.0E-03
      IF (CURRENT(M) .LE. -10.0E-03) CURRENT(M)=-10.0E-03
      VOLT=CURRENT(M)*500.0
C DIGITAL TO ANALOG (D/A) CALIBRATED EQUATION
      Y=NINT(((VOLT+9.9868)/4.8766)*1000.0)
      STATUS=ALDV(0,Y)
      D2OLD=D2
      VOLD=VNEW
      PSIVOD=PSIV
C
  1   M=M+1
      IF ( M .LE. NUM ) GOTO 5
      WRITE(*,*)'Game Over!'
      CALL GETTIM(FH,FM,FS,FSS)
C RESET D/A BOARD
      Y=2048
      STATUS=ALDV(0,Y)
      STATUS=ALTERM()
      WRITE(15,*)'Lambda=',LAMBDA,'R AND G=',R,G,'Payload=',PAYLOAD
      DO 200 M=0,NUM-1
```

```
 200  WRITE(15,25)CURRENT(M+1),PSI(M),PSID(M+1),S(M+1),PHI(M+1)
 25   FORMAT(5F10.4)
      CLOSE(15)
      END
*************************************************************************
C Straight Sliding Mode Control

$INCLUDE:'ATLDEFS.FOR'
$INCLUDE:'ATLERRS.FOR'
      PARAMETER (NUM=1000)
      INTEGER*2 ADGAINS(16),ADCHAN(16),ICONFIG(16),BASEADR,CDEVID,
     +CDEVF,LG,SCAN,DAVAL1,DAVAL2,Y,STATUS
      INTEGER*2 FH,FM,FS,FSS
C
      REAL BHAT,FHAT,L,PI,K,R(3),IHAT
      REAL PSID(NUM),PSIVD(NUM),PSIJERK(NUM),S(NUM),CURRENT(NUM)
      REAL PSI(0:NUM),PSIV(0:NUM),INTGS
      COMMON/CONFIG/ICONFIG
      EQUIVALENCE
      (ICONFIG(KCBASEADR),BASEADR),(ICONFIG(KCDEVID),CDEVID)
     +,(ICONFIG(KCDEVFLAGS),CDEVFLG),(ICONFIG(KCSCAN),SCAN),(ICONFIG(KCC
     +HANNELS),CHAN)
C
      DATA R/300.0,30.0,1.0/,SPEED,L/10.0,0.9985/
      DATA SMALL,SMALFIL/2*0.0/,RATE,FREQ/2000.0,200.0/
      DATA PSIJERK/NUM*0.0/,PSIV(0),D2OLD,VOLD,INTGS/4*0.0/
C UNCERTAINIES
      DATA BHAT,DB/1.1E4,295.0/,FHAT,DF/-5.7732E4,6.7114E4/
C
      PRINT *,'Please Enter Phi AND W (Real Number Please)!'
      READ *,PHI,W
      PRINT *,'Please Enter Desired Trajectory "1"  or "2" '
      READ *,IANS
C DESIRED TRAJECTORY
      IF(IANS .EQ. 1) THEN
        DO 100 M=1,NUM
         TIME=REAL(M)/FREQ
          IF (TIME .LE. 1.0) THEN
             PSID(M)=TIME
             PSIVD(M)=1.0
          ELSE IF((TIME .GE. 1.0) .AND. (TIME .LE. 2.0)) THEN
             PSID(M)=1.0
             PSIVD(M)=0.0
          END IF
```

```
          IF((TIME .GT. 2.0) .AND. (TIME .LE. 2.4)) THEN
             PSID(M)=3.0-TIME
             PSIVD(M)=-1.0
          ELSE IF(TIME .GE. 2.4) THEN
             PSID(M)=0.6
             PSIVD(M)=0.0
          END IF
100     CONTINUE
      ELSE IF(IANS .EQ. 2) THEN
        DO 500 M=1,NUM
         PSID(M)=1.0
         PSIVD(M)=0.0
         PSIJERK(M)=0.0
500     CONTINUE
      END IF
C
      M=0
      PI=4*ATAN(1.0)
      PAR=PI/1054.0
      R1B=1.0/(R(3)*BHAT)
      R1NOM=SQRT(R(1)**2+R(2)**2+R(3)**2)
      OPEN(UNIT=15,FILE='C:\ATLAB\TMP\CTRL2.M',STATUS='NEW')
C INITIALIZE DATA ACQUSITION BOARD
      STATUS=ALINIT()
      STATUS=ALSB(1)
      STATUS=ALSF(RATE)
      STATUS=ALRSET()
      STATUS=ALGC(ICONFIG)
      STATUS=ALDV(0,Y)
C
      WRITE(*,*)' Are you ready (Ctrl-C) ? '
      WRITE(*,*)' When you are ready to go, press "1" and "return" '
      READ *,IANS1
      IF(IANS1 .EQ. 1) GOTO 7
C BEGIN SIGNAL PROCESSING
    7 CALL GETTIM(IH,IM,IS,ISS)
    5 STATUS=ALAV(1,1,DAVAL1)
      STATUS=ALAV(2,1,DAVAL2)
      D1=REAL(DAVAL1-2048)
      D2=REAL(DAVAL2-2048)
C FILTERED VERSION OF SMALL MOTION SIGNAL COMING OUT FROM STRAIN
INDICATOR
      SMALFIL=0.94175*SMALFIL+0.029129*(D2OLD+D2)
C CALIBRATED EQUATION BETWEEN DISPLACEMENT(cm) AND SMALFIL
```

```
      SMALL=-1.0*(SMALFIL+15.0)
      SMANG=SMALL/(100.0*L)
C LARGE MOTION CALIBRATED EQUATION
      THETA=PAR*(D1-30.0)
C TOTAL ANGLE
      PSI(M)=THETA+SMANG
C CONTROLLER DESIGN
      IF(M .EQ. 0) GOTO 1
      POSERR=PSI(M)-PSID(M)
C FILTERED VERSION OF TOTAL VELOCITY, PSIV(M)
      VNEW=(PSI(M)-PSI(M-1))*FREQ
      PSIV(M)=0.975309*PSIV(M-1)+0.01234568*(VOLD+VNEW)
      VELERR=PSIV(M)-PSIVD(M)
      S(M)=R(1)*POSERR+R(2)*VELERR
      INTGS=INTGS+(S(M)/FREQ)
C NORMINAL VALUE
      IHAT=R(1)*(-VELERR)+R(3)*(PSIJERK(M)-F[illegible]-((W**2)*INTGS)
C TO FIND K
      K=(R1NOM*(DF+DB*(R1B*IHA[illegible])+SPEED)/(1.0-(DB*R1B))
      LAMBDA=(SQRT(1+DB*R1B))*W
      PHI(M)=(K/(2*ZETA*LAMbDA))*(DB*R1B)
      STD=S(M)/PHI(M)
      IF(ABS(S(M)) .LT. PHI(M)) SAT=STD
      IF(ABS(S(M)) .GE. PHI(M)) THEN
        IF(STD .GE. 0.0) SAT=1.0
        IF(STD .LT. 0.0) SAT=-1.0
      ENDIF
      CURRENT(M)=R1B*((IHAT - K*SAT)/1000.0)
      IF (CURRENT(M) .GE. 10.0E-03) CURRENT(M)=10.0E-03
      IF (CURRENT(M) .LE. -10.0E-03) CURRENT(M)=-10.0E-03
      VOLT=CURRENT(M)*500.0
C DIGITAL TO ANALOG (D/A) CALIBRATED EQUATION
      Y=NINT(((VOLT+9.9868)/4.8766)*1000.0)
      STATUS=ALDV(0,Y)
      D2OLD=D2
      VOLD=VNEW
C
  1   M=M+1
      IF ( M .LE. NUM ) GOTO 5
      WRITE(*,*)'Game Over!'
      CALL GETTIM(FH,FM,FS,FSS)
C RESET D/A BOARD AND TERMINATION
      Y=2048
      STATUS=ALDV(0,Y)
```

```
      STATUS=ALTERM()
      WRITE(15,*)'%R=',R
      WRITE(15,*)'%PHI=',PHI,'    W=',W
      DO 200 M=0,NUM-1
200   WRITE(15,25)CURRENT(M+1),PSI(M),PSID(M+1),PSIV(M),S(M+1)
25    FORMAT(1X,5F12.4)
      CLOSE(15)
      END
*************************************************************************

C Versatile Sliding Mode Control
$INCLUDE:'ATLDEFS.FOR'
$INCLUDE:'ATLERRS.FOR'
      PARAMETER (NUM=1000)
      INTEGER*2 ADGAINS(16),ADCHAN(16),ICONFIG(16),BASEADR,CDEVID,
     +CDEVF,LG,SCAN,DAVAL1,DAVAL2,Y,STATUS
      INTEGER*2 FH,FM,FS,FSS
C
      REAL BHAT,FHAT,L,PI,K,R(3),G,IHAT
      REAL PSID(NUM),PSIVD(NUM),PSIJERK(NUM),SDOT(NUM),CURRENT(NUM)
      REAL PSI(0:NUM),PSIV(0:NUM),INTGS,INTGERR
      COMMON/CONFIG/ICONFIG
      EQUIVALENCE
      (ICONFIG(KCBASEADR),BASEADR),(ICONFIG(KCDEVID),CDEVID)
     +,(ICONFIG(KCDEVFLAGS),CDEVFLG),(ICONFIG(KCSCAN),SCAN),(ICONFIG(KCC
     +HANNELS),CHAN)
C
      DATA R,G/300.0,30.0,1.0,1000.0/SPEED,L/10.0,0.9985/
      DATA SMALL,SMALFIL,INTGERR/3*0.0/,RATE,FREQ/2000.0,200.0/
      DATA PSIJERK/NUM*0.0/,PSIV(0),S,D2OLD,VOLD,INTGS/5*0.0/
      DATA SDOTO,SDOTN/2*0.0/
C UNCERTAINIES
      DATA BHAT,DB/1.1E4,295.0/,FHAT,DF/-5.7732E4,6.7114E4/
C
      PRINT *,'Please Enter Phi, Wn, AND Z (Real Number Please)!'
      READ *,PHI,W,Z
      PRINT *,'Please Enter Desired Trajectory "1"  or "2" '
      READ *,IANS
C DESIRED TRAJECTORY
      IF(IANS .EQ. 1) THEN
        DO 100 M=1,NUM
         TIME=REAL(M)/FREQ
          IF (TIME .LE. 1.0) THEN
             PSID(M)=TIME
```

```
            PSIVD(M)=1.0
          ELSE IF((TIME .GE. 1.0) .AND. (TIME .LE. 2.0)) THEN
            PSID(M)=1.0
            PSIVD(M)=0.0
          END IF
          IF((TIME .GT. 2.0) .AND. (TIME .LE. 2.4)) THEN
            PSID(M)=3.0-TIME
            PSIVD(M)=-1.0
          ELSE IF(TIME .GE. 2.4) THEN
            PSID(M)=0.6
            PSIVD(M)=0.0
          END IF
 100    CONTINUE
      ELSE
        DO 300 M=1,NUM
         TIME=REAL(M)/FREQ
         PSID(M)=1.0
         PSIVD(M)=0.0
C          PSIAD(M)=0.0
         PSIJERK(M)=0.0
 300    CONTINUE
      ENDIF
C
      M=0
      PI=4*ATAN(1.0)
      PAR=PI/1054.0
      R1B=1.0/(R(3)*BHAT)
      R1NOM=SQRT(R(1)**2+R(2)**2+R(3)**2)
      OPEN(UNIT=15,FILE='C:\ATLAB\TMP\CTRL3.M',STATUS='NEW')
C INITIALIZE DATA ACQUSITION BOARD
      STATUS=ALINIT()
      STATUS=ALSB(1)
      STATUS=ALSF(RATE)
      STATUS=ALRSET()
      STATUS=ALGC(ICONFIG)
      STATUS=ALDV(0,Y)
C
      WRITE(*,*)' Are you ready (Ctrl-C) ? '
      WRITE(*,*)' When you are ready to go, press "1" and "return" '
      READ *,IANS1
      IF(IANS1 .EQ. 1) GOTO 7
C BEGIN SIGNAL PROCESSING
  7  CALL GETTIM(IH,IM,IS,ISS)
  5  STATUS=ALAV(1,1,DAVAL1)
```

```
      STATUS=ALAV(2,1,DAVAL2)
      D1=REAL(DAVAL1-2048)
      D2=REAL(DAVAL2-2048)
C FILTERED VERSION OF SMALL MOTION SIGNAL COMING OUT FROM STRAIN
INDICATOR
      SMALFIL=0.94175*SMALFIL+0.029129*(D2OLD+D2)
C CALIBRATED EQUATION BETWEEN DISPLACEMENT(cm) AND SMALFIL
      SMALL=-1.0*(SMALFIL+15.0)
      SMANG=SMALL/(100.0*L)
C LARGE MOTION CALIBRATED EQUATION
      THETA=PAR*(D1-30.0)
C TOTAL ANGLE
      PSI(M)=THETA+SMANG
C CONTROLLER DESIGN (VERSATILE)
      IF(M .EQ. 0) GOTO 1
      POSERR=PSI(M)-PSID(M)
      INTGERR=INTGERR + POSERR/FREQ
C FILTERED VERSION OF TOTAL VELOCITY, PSIV(M)
      VNEW=(PSI(M)-PSI(M-1))*FREQ
      PSIV(M)=0.975309*PSIV(M-1)+0.01234568*(VOLD+VNEW)
      VELERR=PSIV(M)-PSIVD(M)
      SDOTN=R(1)*POSERR+R(2)*VELERR+G*INTGERR-Z*S
      SDOT(M)=0.94175*SDOT(M-1)+0.029129*(SDOTO+SDOTN)
      S=S+SDOT(M)/FREQ
C NORMINAL VALUE
      IHAT=R(1)*(-VELERR)+R(3)*(PSIJERK(M)-FHAT)-G*POSERR+Z*SDOT(M)
     +-((W**2)*S)
C TO FIND K
      K=(R1NOM*(DF+DB*(R1B*IHAT))+SPEED)/(1.0-(DB*R1B))
      LAMBDA=(SQRT(1+DB*R1B))*W
      PHI(M)=(K*(DB*R1B))/((2*ZETA*LAMBDA)+(DB*R1B*Z))
C
      STD=SDOT(M)/PHI(M)
      IF(ABS(SDOT(M)) .LT. PHI(M)) SAT=STD
      IF(ABS(SDOT(M)) .GE. PHI(M)) THEN
        IF(STD .GE. 0.0) SAT=1.0
        IF(STD .LT. 0.0) SAT=-1.0
      ENDIF
      CURRENT(M)=R1B*((IHAT - K*SAT)/1000.0)
      IF (CURRENT(M) .GE. 10.0E-03) CURRENT(M)=10.0E-03
      IF (CURRENT(M) .LE. -10.0E-03) CURRENT(M)=-10.0E-03
      VOLT=CURRENT(M)*500.0
C DIGITAL TO ANALOG (D/A) CALIBRATED EQUATION
      Y=NINT(((VOLT+9.9868)/4.8766)*1000.0)
```

```
      STATUS=ALDV(0,Y)
      D2OLD=D2
      VOLD=VNEW
      SDOTO=SDOTN
C
  1   M=M+1
      IF ( M .LE. NUM ) GOTO 5
      WRITE(*,*) 'Game Over !'
      CALL GETTIM(FH,FM,FS,FSS)
C RESET D/A BOARD AND TERMINATION
      Y=2048
      STATUS=ALDV(0,Y)
      STATUS=ALTERM()
      WRITE(15,*)'%R=',R
      WRITE(15,*)'%G=',G
      WRITE(15,*)'%PHI= ',PHI,'  W= ',W,'  Z= ',Z
      DO 200 M=0,NUM-1
 200  WRITE(15,25)CURRENT(M+1),PSI(M),PSID(M+1),PSIV(M),SDOT(M+1)
 25   FORMAT(1X,5F12.4)
      CLOSE(15)
      END
```

LIST OF REFERENCES

1. Chang, L.W., *Dynamic Analysis of Robotic Manipulators with Flexible Links*, Ph.D. Dissertation, School of Mechanical Engineering, Purdue University, W. Lafayette, IN. 1984.

2. Chang, L.W., and Hamilton J.F., " A Sequential Integration Method," *Journal of Dynamics Systems, Measurement, and Control*, vol.110, pp. 383-388, December 1988.

3. Petroka, R.P., *Computer Simulation and Experimental Validation of a Dynamic Model (Equivalent Rigid Link System) on a Single Link Flexible Manipulator*, Master's Thesis, Naval Postgraduate School, Monterey, California, October 1982.

4. Chang, L.W., and Gannon K.P., "A Dynamic Model on a Single-Link Flexible Manipulator," *Journal of Vibration and Acoustics*, vol.112, pp. 138-143, January 1990.

5. Kirkland, M., *Implementation of Dynamic Control of a Single-Link Arm Using a General Micro-Computer*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1988.

6. DeCarlo, R.A., et. al., " Variable Structure Control of Nonlinear Multivariable Systems: A Tutorial," *Proceedings of the IEEE*, vol.76 No.3, March 1988.

7. White, B.A., "Reduced-Order Switching Functions in Variable-Structure Control Systems," *IEE PROC.*, vol.130, Pt. D, no.2, March 1983.

8. Utkin, V.I., "Survey Paper- Variable Structure Systems with Sliding Mode," *IEEE Trans. Automatic Control*, vol, AC-22, no.2, pp.212-222, 1977.

9. Slotine, J.J., and Sastry, S.S., "Tracking Control of Non-Linear Systems Using Sliding Surfaces, with Application to Robot Manipulators," *Int. Journal of Control*, vol. 38, no. 2, pp.465-492, 1983.

10. Fan, T.C., *A Robust Control of a Vertical-Plane Motion for a Electrohydraulically-Actuated Signal-Flexible-Link Arm*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1989.

11. Chang, L.W., "A Straight Sliding Control with a First-Order Plus Integral Sliding Condition," submitted to *IEEE Trans. on Automatic Control*, 1990.

12. Chang, L.W., "A Versatile Sliding Control with a Second-Order Sliding Condition," submitted to *ASME Journal of Dynamics Systems, Measurement and Control*, 1990.

13. Merrit, H.E., *Hydraulic Control Systems*, John Wiley & Sons, Inc., 1967.

14. Park, K.S., *Control System Simulation for a Single-Link Flexible Arm*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1987.